The Astrophysical Journal Supplement Series, 253:21 (44pp), 2021 March https://doi.org/10.3847/1538-4365/abd700

thornado-hydro: A Discontinuous Galerkin Method for Supernova Hydrodynamics with Nuclear Equations of State*

David Pochik[1], Brandon L. Barker[2,5], Eirik Endeve[1,3], Jesse Buffaloe[1], Samuel J. Dunham[1,4], Nick Roberts[1], and Anthony Mezzacappa[1]
[1] Department of Physics and Astronomy, University of Tennessee, Knoxville, TN 37996, USA; endevee@ornl.gov
[2] Department of Physics and Astronomy, Michigan State University, East Lansing, MI 48824, USA
[3] Computer Science and Mathematics Division, Oak Ridge National Laboratory, Oak Ridge, TN 37831, USA
[4] Department of Physics and Astronomy, Vanderbilt University, Nashville, TN 37235, USA
Received 2020 October 23; revised 2020 December 19; accepted 2020 December 21; published 2021 March 10

Abstract

This paper describes algorithms for nonrelativistic hydrodynamics in the toolkit for high-order neutrino radiation hydrodynamics (thornado), which is being developed for multiphysics simulations of core-collapse supernovae (CCSNe) and related problems with Runge–Kutta discontinuous Galerkin (RKDG) methods. More specifically, thornado employs a spectral-type nodal collocation approximation, and we have extended limiters—a slope limiter to prevent nonphysical oscillations and a bound-enforcing limiter to prevent nonphysical states—from the standard RKDG framework to be able to accommodate a tabulated nuclear equation of state (EoS). To demonstrate the efficacy of the algorithms with a nuclear EoS, we first present numerical results from basic test problems in idealized settings in one and two spatial dimensions, employing Cartesian, spherical-polar, and cylindrical coordinates. Then, we apply the RKDG method to the problem of adiabatic collapse, shock formation, and shock propagation in spherical symmetry, initiated with a 15 $M_\odot$ progenitor. We find that the extended limiters improve the fidelity and robustness of the RKDG method in idealized settings. The bound-enforcing limiter improves the robustness of the RKDG method in the adiabatic collapse application, while we find that slope limiting in characteristic fields is vulnerable to structures in the EoS—more specifically, in the phase transition from nuclei and nucleons to bulk nuclear matter. The success of these applications marks an important step toward applying RKDG methods to more realistic CCSN simulations with thornado in the future.

Unified Astronomy Thesaurus concepts: Computational methods (1965); Core-collapse supernovae (304); Hydrodynamical simulations (767); Nuclear astrophysics (1129)

1. Introduction

Stars with zero-age main sequence (ZAMS) masses $M_{\rm ZAMS} \gtrsim 8M_\odot$ end their lives as spectacular explosions known as core-collapse supernovae (CCSNe). These explosions are at the heart of some of the most important questions in astrophysics. They are the primary catalysts of galactic chemical evolution, producing and dispersing many of the elements heavier than hydrogen and helium, and provide feedback into the interstellar medium. They may even be a source of the lighter first peak *r*-process elements (Martínez-Pinedo et al. 2014), though neutron star mergers are likely the primary production site for the *r*-process (Kasen et al. 2017). Their cores are the foundries for compact objects including those recently detected by Advanced LIGO and Virgo (Abbott et al. 2016, 2017a, 2017b, 2020). Through their observables and the compact objects left behind, we may even begin to probe the nature of nuclear matter (Schneider et al. 2019).

Throughout their lives, these massive stars undergo successive cycles of nuclear fusion, forging heavier elements in their cores. At the end of a star's lifetime, fusion processes build up a degenerate iron core that is unable to undergo nuclear fusion itself. This iron core, supported thus far by electron degeneracy pressure, grows to the effective Chandrasekhar mass (Baron & Cooperstein 1990) and, no longer able to balance gravity, subsequently collapses. During the collapse, runaway electron capture processes accelerate the collapse and produce vast numbers of neutrinos, while photodissociation of iron- group nuclei robs the core of more energy. Eventually, the core reaches nuclear density and the nuclear strong force becomes repulsive, effectively stiffening the equation of state (EoS) tremendously, and collapse is halted in the inner core. The collapse rebounds and produces a strong shock that is driven through the outer core. Ultimately, through a combination of neutrino cooling and dissociation of iron-group nuclei, the shock runs out of energy and stalls before escaping the core, becoming an accretion shock. Meanwhile, the inner core regains equilibrium in the form of a newborn proto-neutron star (PNS).

Providing a mechanism to revive the stalled shock and drive the explosion is among the forefront questions in the study of CCSNe. Of the proposed mechanisms, the most favored has been the delayed neutrino-driven mechanism (Bethe & Wilson 1985). Neutrinos emitted from the surface of the cooling PNS, aided by hydrodynamic and magnetohydrodynamic instabilities, deposit energy below the stalled shock and reinvigorate the explosion. Of the other proposed mechanisms, the magnetorotational mechanism—wherein a rapidly rotating PNS supplies energy to power the shock

[5] NSF Graduate Research Fellow.

(Akiyama et al. 2003)—has potential, but likely does not account for most CCSNe. A key characteristic of magnetorotationally driven SNe is the formation of collimated jets, which are not seen in the vast majority of supernova remnants (e.g., see Soderberg et al. 2010). Additionally, for this mechanism to be effective the stellar core must be very rapidly rotating, beyond the rotation rates commonly achieved through stellar evolution (Heger et al. 2005). Ultimately, any successful mechanism must not only revive the shock but also explain the observations of supernovae (e.g., light curves and spectra).

For several decades, this was the state of the field. These mechanisms saw little success until relatively recently: spherically symmetric (spatially one-dimensional (1D)) simulations of CCSNe consistently failed to produce explosions. It was not until computing resources allowed for axisymmetric (spatially two-dimensional (2D)), and eventually full-physics three-dimensional (3D), simulations that successful explosions could be consistently produced without modified or parameterized physics. Ultimately, the reason for this is 1D fails to capture the fundamentally nonspherical nature of CCSNe, and hydrodynamic instabilities are unable to develop. The CCSN explosion mechanism has been the subject of decades of work and still remains incompletely described (for in-depth reviews, see, e.g., Bethe 1990; Mezzacappa 2001, 2005; Janka et al. 2012, 2016; Burrows 2013; Hix et al. 2014; Müller et al. 2016; Couch 2017).

Hydrodynamics, along with gravity and neutrino transport, plays a key role in the dynamics of CCSNe. This starts with the progenitors, which in nature are multidimensional and likely involve a complicated mixing of elements in the convectively burning shells (see, e.g., Arnett & Meakin 2011). Further, it has been shown that asphericities in progenitors can mean the difference between a model that explodes and a model that does not (Couch & Ott 2013). However, regardless of the progenitor, after the core rebounds, it is known that the shocked fluid develops instabilities.

Once the bounce shock stalls and the neutrino-heating (or gain) region is established below the shock, at least two hydrodynamical instabilities may contribute to the evolution of the shock: neutrino-driven convection (Herant et al. 1992) and standing accretion shock instability (SASI; Blondin et al. 2003). Both of these instabilities create turbulence in the post-shock flow, and that turbulence contributes ram pressure that enlarges the extent of the gain region (Murphy et al. 2013), thus increasing the efficacy of neutrino heating, therefore aiding the explosion (see Couch & Ott 2015, and references therein). Which effect is more dynamically important, however, may depend on the progenitor mass (Müller et al. 2012; Hanke et al. 2013; Summa et al. 2016; Vartanyan et al. 2019). Regardless of which effect is dominant, simulations should be able to satisfactorily quantify the turbulence, and in particular should be able to capture the turbulent energy cascade from the energy-carrying scale through the inertial scale, down to the (numerical) dissipation scale. However, a consensus has not yet been reached as to what, in terms of angular resolution, is required to adequately capture the turbulent energy cascade. In particular, Radice et al. (2015), Abdikamalov et al. (2015), and Casanova et al. (2020) suggest that resolutions much lower than 1° may be necessary (due to the numerical dissipation of the scheme, which creates a "bottleneck" for energy transfer at a scale set by the scheme), but recently Melson et al. (2020) argued that 1° resolution is sufficient to obtain a clear distinction between the inertial and dissipation scales. Additionally, Endeve et al. (2012) showed that turbulence from the SASI can amplify magnetic fields, and more recently, Müller & Varma (2020) found that turbulently amplified magnetic fields can aid neutrino-driven explosions, even in slowly rotating progenitors. See Radice et al. (2018) for a recent review of turbulence in CCSNe.

In addition to the hydrodynamic instabilities occurring in the shocked mantle, the PNS undergoes convection and potentially other instabilities due to entropy and electron fraction gradients (Bruenn et al. 2004), which has an effect on the luminosity of heavy flavored neutrinos as well as the mean energies of all neutrino flavors (Buras et al. 2006). This may not directly affect the shock dynamics, but it does give rise to the recently discovered lepton number emission self-sustained asymmetry (Tamborra et al. 2014), which may hold implications for the composition of the ejecta. For more detailed discussions on the role of hydrodynamic instabilities in CCSNe, we refer to the recent review by Müller (2020).

Insight into hydrodynamic phenomena can often be gained by treating the fluid as polytropic (in the CCSN context, see, e.g., Yahil 1983; Blondin et al. 2003); i.e., the fluid pressure p is assumed to be proportional to a power law of the mass density ρ, which gives rise to the polytropic EoS, $p \propto \rho^{\Gamma}$, where $\Gamma = \left(\frac{\partial \ln p}{\partial \ln \rho}\right)$ is the adiabatic index.[6] However, relating the state variables by this expression ignores the nuclear interactions and compositions in stellar collapse; e.g., the polytropic EoS fails to capture the response in pressure due to the thermal or compositional changes that are typical in a stellar environment. For the conditions prevalent in stellar interiors, particularly in the high-density regimes of stellar collapse, a simple analytic form for the EoS likely does not exist. Instead, an EoS for this case is often created by minimizing a thermodynamic potential —e.g., the Helmholtz free energy—for a system of particles under stellar conditions (see, e.g., Swesty 1996; Fryxell et al. 2000; Timmes & Swesty 2000). Once the free energy is known, other relevant quantities, such as pressure, internal energy, and entropy, can easily be obtained.

The task of developing an equation of state for realistic CCSN simulations has remained a pertinent objective for several decades. Important contributions toward this effort include the Lattimer & Swesty (1991, LS) and Shen et al. (1998, STOS) EoSs. The LS EoS used a compressible liquid-drop model (see, e.g., Lattimer et al. 1985), while STOS used a relativistic mean-field (RMF) model with the TM1 parameter set (see, e.g., Sugahara & Toki 1994). However, due to the importance of including light nuclei in CCSN simulations, a notable drawback for both the LS and STOS EoSs was their exclusion of all light nuclei other than alpha particles (Hempel et al. 2012; Steiner et al. 2013b). Further advances include the hadronic EoSs from G. Shen (Shen et al. 2011a, 2011b), which build upon the NL3 (Lalazissis et al. 1997) and FSUgold (Todd-Rutel & Piekarewicz 2005) parameter sets. Additionally, unlike the LS and STOS EoSs, the statistical model of Hempel et al. (2012, HS; see also Steiner et al. 2013b) does not use the single-nucleus approximation for heavy nuclei, but includes a more realistic compositional distribution of nuclei.

Moreover, recent neutron star observations (see, e.g., Steiner et al. 2013a; Greif et al. 2020) and observations of other

[6] Contrary to a realistic model, the adiabatic index for a polytropic model remains constant through space and time.

astronomical phenomena (see, e.g., Greif et al. 2020 and references therein), experiments in nuclear physics (see, e.g., Greif et al. 2020), and experiments in relativistic heavy-ion collisions (see, e.g., Oertel et al. 2017 and references therein) have led to the development of multiple EoSs for dense nuclear matter that are applicable to CCSN simulations (see, e.g., Steiner et al. 2013a, 2013b). These equations of state provide thermodynamic quantities as functions of density, temperature, and electron fraction. The SHFo/SFHx EoSs from Steiner et al. (2013a, 2013b) build upon the statistical model used in HS and constrain properties of nucleonic matter with an RMF model (see, e.g., Shen et al. 1998, 2011a, 2011b). The most probable mass–radius relationship derived from neutron star (NS) observations was used to build the "optimal" SFHo EoS, while the "extreme" SFHx EoS is built around a minimized radius model for low-mass NSs (Steiner et al. 2010, 2013a). For our purposes, the importance of these equations of state lies in their ability to resolve various physical regimes in CCSNe, including the phase transition from nuclei and nucleons to bulk nuclear matter at high densities ($\rho \sim 10^{14}$ g cm^{-3}; Steiner et al. 2013b) and the high-density rebound of the core, which determines the initial strength of the shock (Shen et al. 1998). We note that these EoSs do not include lower-density/temperature regimes, i.e., they do not describe matter out of nuclear statistical equilibrium (NSE); but see, e.g., Bruenn et al. (2020) for the treatment of non-NSE regions in CCSN models.

Clearly, multidimensional, multiphysics models of CCSNe require advanced simulation tools and massive computational resources, and to that end, there are several production codes in existence; e.g., Aenus-Alcar (Just et al. 2015), Castro (Almgren et al. 2010), Chimera (Bruenn et al. 2020), CoCoNuT-Vertex (Müller et al. 2010), FLASH (Fryxell et al. 2000; Dubey et al. 2009; O'Connor & Couch 2018), Fornax (Skinner et al. 2019), Prometheus-Vertex (Rampp & Janka 2002), and Zelmani (Ott et al. 2009; Roberts et al. 2016), and the codes of Sumiyoshi & Yamada (2012), Nagakura et al. (2014), and Kuroda et al. (2016). To solve the equations of hydrodynamics —with the aim of capturing shocks and resolving turbulent flows—these codes use variations of either the finite-difference or the finite-volume high-resolution shock-capturing method, in either an Eulerian or semi-Lagrangian framework. In particular, the finite-volume method divides the computational domain into finite cells (or volumes), formulates the hydrodynamics equations in integral form, and solves for physical quantities (e.g., mass density) in terms of cell averages. The cell averages are updated by accounting for (1) fluxes through the surface enclosing each cell and (2) volume sources (e.g., due to gravity). The integral formulation leads naturally to good conservation properties and allows for discontinuous solutions (e.g., shocks). In computing the surface fluxes, local polynomials are reconstructed using cell averages of the local cell and its neighbors. The local polynomials are then used to assign left and right states at each cell interface as inputs to a Riemann solver, which provides the numerical flux. To avoid nonphysical oscillations around shocks, limiters are applied to the reconstructed polynomial to enforce some degree of monotonicity, which can degrade the formal order of accuracy of the hydrodynamics scheme. (We refer to the above citations for further details on the hydrodynamics algorithms implemented in the specific codes listed.)

As discussed above, turbulence is ubiquitous in the supernova environment and plays a role in the explosion mechanism. It is therefore desirable to maintain good spectral resolution to resolve as much of the turbulent spectrum as possible for a given spatial resolution, and this motivates the use of accurate Riemann solvers and high-order methods. On the other hand, due to their multiphysics nature, CCSN simulations with neutrino transport are computationally expensive, and must run efficiently on distributed memory architectures; e.g., using message passing interface (MPI). Furthermore, because of the high number of degrees of freedom involved in neutrino transport computations (a momentum space is attached to each spatial point), memory limitations require the number of spatial cells assigned to any given MPI process to not be large. For a code to scale well, the number of ghost cells should be limited relative to the number of compute cells to manage the communication overhead, because each MPI process will have a halo region composed of ghost cells populated with data from neighboring processes. While finite-difference and finite-volume methods can achieve high-order accuracy, the computational stencil width increases with increasing order of accuracy, thereby increasing the size of the halo region and the ratio of ghost cells to compute cells, thus impeding good scalability (e.g., Miller & Schnetter 2017).

The discontinuous Galerkin (DG) method (e.g., Cockburn & Shu 2001) is an alternative approach to solving the system of hydrodynamics equations (and many other systems). Similar to finite-volume methods, DG methods divide the computational domain into cells (or elements) and formulate the equations in integral form. However, contrary to finite-volume and finite-difference methods, in the DG method, the solution is approximated by a local polynomial within each element, which implies that more local information is tracked in the solution process (i.e., not just the cell average). Because the full polynomial representation in each element is evolved, the reconstruction step needed in the finite-volume approach is not necessary. Meanwhile, Riemann solvers developed in the context of finite-volume methods can readily be used with DG methods to evaluate numerical fluxes on element interfaces. The DG method is a finite-element method, but does not demand continuity of the local polynomial approximation across element boundaries, and consequently, is well suited to capture shocks and other discontinuities. To prevent nonphysical oscillations in the vicinity of a discontinuity, limiters are applied to the local polynomial to enforce monotonicity. More recently, so-called structure-preserving discretizations, which maintain fundamental physical properties of the system under consideration (e.g., positive mass density and pressure), have been developed within the DG framework (e.g., Zhang & Shu 2011). Another advantage offered by the DG method is high-order spatial accuracy on a compact stencil. Only information from nearest neighbors is needed, independent of the order of accuracy. This makes the DG method well suited for application on massively parallel architectures, because increasing the order of accuracy does not increase the communication overhead as much as other high-order methods (e.g., Miller & Schnetter 2017). The desired combination of shock-capturing capabilities, high-order accuracy in smooth flows, and good scalability make DG methods an appealing choice. Additionally, DG methods are also amenable to *hp*-adaptivity (Remacle et al. 2003), wherein refinement of either the spatial mesh (*h*-refinement) or the local degree of the polynomial approximation (*p*-refinement) can be used to improve the accuracy of the method near shocks while maintaining high-order accuracy in regions of smooth flow. DG methods are also

well suited for problems involving curvilinear coordinates (Teukolsky 2016).

The DG method was introduced already in the 1970s by Reed & Hill (1973) to solve the steady-state neutron transport equation, and the initial framework for solving time-dependent problems with explicit Runge–Kutta time integration (commonly referred to as RKDG methods) was established in a series of papers by Cockburn & Shu (Cockburn & Shu 1989, 1991, 1998; Cockburn et al. 1989, 1990). Today, DG methods are widely used in science and engineering applications and are rapidly gaining popularity in the computational astrophysics community (see, e.g., Radice & Rezzolla 2011; Schaal et al. 2015; Teukolsky 2016; Kidder et al. 2017; Fambri et al. 2018, and references therein), but have so far not been applied to multiphysics CCSN simulations.

The toolkit for high-order neutrino radiation hydrodynamics[7] (`thornado`) is being developed with the goal of realizing multiphysics simulations of CCSNe and related problems with high-order methods. To this end, the hydrodynamics and neutrino transport algorithms in `thornado` are based on the DG method (see, e.g., Chu et al. 2019; Endeve et al. 2019; Laiu et al. 2020). It should be noted that, in addition to exhibiting favorable parallel scalability, DG methods are also an attractive choice for discretizing the neutrino transport equations because they recover the correct asymptotic behavior in the so-called diffusion limit (e.g., Larsen & Morel 1989; Adams 2001), which is characterized by frequent neutrino–matter interactions. Then, because the matter and neutrinos are strongly coupled in the CCSN environment, employing the DG method also for the hydrodynamics is most natural, as this enables the treatment of the coupled physics in a unified mathematical framework. Currently, `thornado` is being developed as a collection of modules, focusing on single-node performance for updating structured data blocks using CPUs and/or GPUs, with the future aim of leveraging an external framework—e.g., AMReX[8] (Zhang et al. 2019)—to support mesh adaptivity.

This paper describes the DG algorithms for nonrelativistic hydrodynamics in `thornado`. We adapt a three-covariant formalism that is sufficiently general to accommodate Cartesian, spherical-polar, and cylindrical spatial coordinates. Although we presented preliminary results obtained with similar algorithms for nonrelativistic and relativistic hydrodynamics in the context of an ideal EoS in Endeve et al. (2019), this paper provides a more comprehensive description of the methods in `thornado`, and, more important, develops the algorithms further in order to accommodate a nuclear EoS. Introducing a nuclear matter EoS leads to more realistic models but also complicates the numerical procedure. For instance, when solving the conservation equations for mass, momentum, and energy, the implementation of a nuclear EoS requires an additional conservation law for electrons, (see, e.g., Colella & Glaz 1985; Zingale & Katz 2015, for similar modifications). Moreover, on-the-fly numerical evaluation of a realistic EoS is computationally expensive (Swesty 1996); thus, for computational expediency, EoSs are provided in tabulated form, and interpolations are used to access quantities away from table vertices, where a thermodynamically consistent interpolation scheme may be required (see, e.g., Swesty 1996; Fryxell et al. 2000; Timmes & Swesty 2000, for a discussion of such interpolation schemes). To limit the scope of this paper, we exclusively consider the SFHo EoS (Steiner et al. 2013a), which is provided in tabulated form by CompOSE.[9] In `thornado`, the interface to the tabulated EoS is through the WeakLib library,[10] which provides auxiliary functionality needed for computations (e.g., input/output and interpolation). As such, the EoS is currently treated as a black box.

The Euler equations in curvilinear coordinates, extended to accommodate a nuclear EoS and self-gravity, are listed in Section 2. Then, in Section 3, we present the RKDG method in `thornado`. Sections 3.1 and 3.2 provide the spatial and temporal discretizations, respectively, which are based on the standard framework from Cockburn & Shu (2001). More specifically, we employ a nodal DG method (e.g., Hesthaven & Warburton 2008) and adopt the spectral-type nodal collocation approximation investigated by Bassi et al. (2013). Sections 3.3 and 3.4 discuss the slope limiter (to prevent nonphysical oscillations) and the bound-enforcing limiter (to prevent nonphysical states), respectively. The extension of these limiters to the case with a tabulated nuclear EoS is nontrivial. First, because slope limiting is most effective when applied to characteristic variables, we provide the characteristic decomposition of the flux Jacobian matrices for a nuclear EoS (Appendix A). Second, because the domain of validity of the nuclear EoS is more complex than the ideal case, we develop an enhanced version of the bound-enforcing limiter of Zhang & Shu (2010a). Section 3.5 describes the Poisson solver for use in spherically symmetric problems with self-gravity, which uses the finite-element method. Section 3.6 provides details on the interpolation methods used to evaluate the tabulated EoS. We use basic trilinear interpolation, which is commonly employed in supernova simulation codes (e.g., Bruenn et al. 2020). In Section 4, to demonstrate the efficacy of the algorithms, we present numerical results from basic test problems (advection and Riemann problems) in idealized settings in one and two spatial dimensions. We also include a test of the Poisson solver. Then, in Section 5, we apply the DG method to the problem of adiabatic collapse, shock formation, and shock propagation in spherical symmetry, using a $15\,M_\odot$ progenitor. Here we focus on aspects of the limiters, resolution dependence, and total energy conservation. Our major goals in this paper are to (1) present the key algorithmic components of the hydrodynamics in `thornado`, (2) assess the implementation given the initial set of algorithmic choices, and (3) identify potential areas for improvement. This will clear the way for incorporating DG methods for neutrino transport and future neutrino radiation-hydrodynamics simulations with `thornado`.

2. Physical Model

2.1. Euler Equations

In this paper, we adopt the nonrelativistic Euler equations of gas dynamics in a coordinate basis (e.g., Rezzolla & Zanotti 2013), supplemented with a nuclear equation of state (EoS), which are given by the mass conservation equation

$$\partial_t \rho + \frac{1}{\sqrt{\gamma}}\,\partial_i(\,\sqrt{\gamma}\,\rho\,v^i\,) = 0, \tag{1}$$

[7] https://github.com/endeve/thornado
[8] https://amrex-codes.github.io
[9] https://compose.obspm.fr
[10] https://github.com/starkiller-astro/weaklib

Table 1
Metric Quantities for Cartesian, Cylindrical, and Spherical-polar Coordinate Systems

Coordinates	x^1	x^2	x^3	h_1	h_2	h_3	γ_{11}	γ_{22}	γ_{33}	$\sqrt{\gamma}$	$\frac{1}{\gamma_{22}}\frac{\partial\gamma_{22}}{\partial x^1}$	$\frac{1}{\gamma_{33}}\frac{\partial\gamma_{33}}{\partial x^1}$	$\frac{1}{\gamma_{33}}\frac{\partial\gamma_{33}}{\partial x^2}$
Cartesian	x	y	z	1	1	1	1	1	1	1	0	0	0
Cylindrical	R	z	ϕ	1	1	R	1	1	R^2	R	0	$2/R$	0
Spherical	r	θ	ϕ	1	r	$r\sin\theta$	1	r^2	$r^2\sin^2\theta$	$r^2\sin\theta$	$2/r$	$2/r$	$2\cot\theta$

the momentum equation

$$\partial_t(\rho\, v_j) + \frac{1}{\sqrt{\gamma}}\,\partial_i(\sqrt{\gamma}\,\Pi^i_{\ j}) = \frac{1}{2}\,\Pi^{ik}\,\partial_j\gamma_{ik} - \rho\,\partial_j\Phi, \tag{2}$$

the energy equation

$$\partial_t E + \frac{1}{\sqrt{\gamma}}\,\partial_i(\sqrt{\gamma}\,[E+p]\,v^i) = -\rho\,v^i\,\partial_i\Phi, \tag{3}$$

and the electron conservation equation

$$\partial_t D_{\rm e} + \frac{1}{\sqrt{\gamma}}\,\partial_i(\sqrt{\gamma}\,D_{\rm e}\,v^i) = 0, \tag{4}$$

where ρ represents the mass density, v^i the components of the fluid three-velocity, $\Pi^i_{\ j} = \rho\,v^i\,v_j + p\,\delta^i_{\ j}$ the stress tensor, p the fluid pressure, $D_{\rm e} = \rho Y_{\rm e}$, where $Y_{\rm e}$ is the electron fraction, $E = \epsilon\rho + \frac{1}{2}\rho v^2$ the total fluid energy density (internal plus kinetic), and ϵ the specific internal energy. The Euler equations are closed with the EoS, where the pressure and specific internal energy are given functions of density, temperature T, and the electron fraction; e.g., $p = p(\rho,\ T,\ Y_{\rm e})$. Thus, Equation (4) is necessary for the inclusion of a nuclear EoS. (Unless stated otherwise, we use the Einstein summation convention where repeated Latin indices run from 1 to 3.) Included on the right-hand sides of Equations (2) and (3) are gravitational sources from the Newtonian gravitational potential Φ, which is obtained from the Poisson equation

$$\frac{1}{\sqrt{\gamma}}\partial_i(\sqrt{\gamma}\,\gamma^{ij}\partial_j\Phi) = 4\pi\,G\,\rho, \tag{5}$$

where G is Newton's constant.

The use of curvilinear coordinates is enabled through the spatial metric tensor γ_{ik}, which gives the squared proper spatial interval

$$ds^2_{\boldsymbol{x}} = \gamma_{ik}\,dx^i\,dx^k. \tag{6}$$

The determinant of the spatial metric is denoted γ. The metric tensor is also used to raise and lower indices on vectors and tensors, e.g., $v_i = \gamma_{ik}\,v^k$. In this paper, we only consider the commonly adopted Cartesian, cylindrical, and spherical-polar coordinate systems (see Table 1 for relevant quantities associated with each of these systems). Thus, the metric tensor is diagonal, and we assume that it is time independent. Note that we also list the scale factors h_1, h_2, and h_3 in Table 1. By specifying the scale factors, components of the spatial metric are obtained from $\gamma_{11} = h_1h_1$, $\gamma_{22} = h_2h_2$, and $\gamma_{33} = h_3h_3$, and the square root of the metric determinant is $\sqrt{\gamma} = h_1h_2h_3$.

For the discussion of the numerical method in Section 3, we rewrite Equations (1)–(4) in a more convenient way as a system of hyperbolic balance equations:

$$\partial_t\boldsymbol{U} + \frac{1}{\sqrt{\gamma}}\,\partial_i(\sqrt{\gamma}\,\boldsymbol{F}^i(\boldsymbol{U})) = \boldsymbol{S}(\boldsymbol{U},\,\Phi), \tag{7}$$

where

$$\boldsymbol{U} = \begin{bmatrix}\rho\\ \rho v_j\\ E\\ D_{\rm e}\end{bmatrix},\quad \boldsymbol{F}^i(\boldsymbol{U}) = \begin{bmatrix}\rho\,v^i\\ \Pi^i_{\ j}\\ (E+p)v^i\\ D_{\rm e}\,v^i\end{bmatrix},\quad \text{and}$$
$$\boldsymbol{S}(\boldsymbol{U},\,\Phi) = \begin{bmatrix}0\\ \frac{1}{2}\,\Pi^{ik}\,\partial_j\gamma_{ik} - \rho\,\partial_j\Phi\\ -\rho\,v^i\,\partial_i\Phi\\ 0\end{bmatrix} \tag{8}$$

are the vector of the evolved quantities, the flux vectors, and the source vector, respectively. We split the source vector further as $\boldsymbol{S}(\boldsymbol{U},\,\Phi) = \boldsymbol{S}^{\gamma}(\boldsymbol{U}) + \boldsymbol{S}^{\Phi}(\boldsymbol{U},\,\Phi)$, where

$$\boldsymbol{S}^{\gamma}(\boldsymbol{U}) = \begin{bmatrix}0\\ \frac{1}{2}\,\Pi^{ik}\,\partial_j\gamma_{ik}\\ 0\\ 0\end{bmatrix}\quad\text{and}\quad \boldsymbol{S}^{\Phi}(\boldsymbol{U},\,\Phi) = -\rho\begin{bmatrix}0\\ \partial_j\Phi\\ v^i\,\partial_i\Phi\\ 0\end{bmatrix}. \tag{9}$$

2.2. Equation of State

The EoS provides thermodynamic quantities such as pressure, internal energy, and entropy (dependent variables) as functions of the independent variables, e.g., density, temperature, and electron fraction. (Other choices for the independent variables—e.g., density, entropy, and electron fraction—are of course also possible, but in the nuclear astrophysics modeling community, it is perhaps most common to use ρ, T, and $Y_{\rm e}$.) These dependent variables, and in some cases their derivatives, are crucial for modeling hydrodynamics, nuclear reactions, and neutrino transport in CCSNe. Of particular importance for numerical methods for hydrodynamics is the relationship between the EoS and the well posedness of the system given by Equation (7). Specifically, the system is said to be hyperbolic if the Jacobian matrices $\partial\boldsymbol{F}^i/\partial\boldsymbol{U}$ can be diagonalized with a set of real eigenvalues $\{\lambda^i_1,\ldots,\lambda^i_6\}$ and has a set of linearly independent right eigenvectors $\{\boldsymbol{r}^i_1,\ldots,\boldsymbol{r}^i_6\}$ such that (see LeVeque 1992; Rezzolla & Zanotti 2013)

$$(\partial\boldsymbol{F}^i/\partial\boldsymbol{U})\boldsymbol{r}^i_j = \lambda^i_j\,\boldsymbol{r}^i_j,\quad \text{for}\quad j = 1,\ldots,6. \tag{10}$$

(In Equation (10), repeated indices do not imply summation, but rather that it must hold for each of the three flux vectors.) For the system in Equation (7), the eigenvalues are given by

$\{v^i - c_s\sqrt{\gamma^{ii}}, v^i, v^i, v^i, v^i, v^i + c_s\sqrt{\gamma^{ii}}\}$, where c_s is the sound speed: $c_s^2 = (\partial p/\partial \rho)_{s,Y_e}$, where s is the entropy per baryon. A fundamental property of hyperbolic equations is that they are well posed, which makes them suitable for a numerical solution (see, e.g., Rezzolla & Zanotti 2013 for a discussion). Thus, a necessary condition for our system to be suitable for a numerical solution is $c_s^2 > 0$. When the independent variables are chosen to be ρ, T, and Y_e, the square of the sound speed can be written explicitly in terms of thermodynamic derivatives as

$$c_s^2 = \left(\frac{\partial p}{\partial \rho}\right)_{s,Y_e} = \left(\frac{\partial p}{\partial \rho}\right)_{T,Y_e} - \left(\frac{\partial s}{\partial T}\right)_{\rho,Y_e}^{-1} \times \left(\frac{\partial p}{\partial T}\right)_{\rho,Y_e}\left(\frac{\partial s}{\partial \rho}\right)_{T,Y_e}. \quad (11)$$

The sound speed, or a related quantity, is typically included with a tabulated EoS. In addition, advanced numerical methods make use of the eigenvectors in Equation (10), e.g., for the characteristic limiting described in Section 3.3. These eigenvectors in turn depend on additional thermodynamic derivatives, whose estimation from the EoS table is discussed in Section 3.6. For use in computations, `thornado` has been developed to use the EoS infrastructure provided by the WeakLib library. (Specifically, WeakLib supplies trilinear interpolation and derivatives computed by analytic differentiation of the trilinear interpolation formula.)

3. Numerical Method

3.1. Discontinuous Galerkin Method

In `thornado`, we employ the Runge–Kutta discontinuous Galerkin (RKDG) method to solve the Euler equations given by Equation (7). (We refer to Cockburn & Shu 2001 for an excellent review on the RKDG method and to Shu 2016 for a summary of more recent developments.) To this end, the d-dimensional computational domain $D \subset \mathbb{R}^d$ is subdivided into the union $\mathcal{T}$ of nonoverlapping elements $\boldsymbol{K}$ such that $D = \cup_{\boldsymbol{K}\in\mathcal{T}}\boldsymbol{K}$. We take each element to be a logically Cartesian box

$$\boldsymbol{K} = \{\boldsymbol{x}: x^i \in K^i \coloneqq (x_L^i, x_H^i),\ i = 1,\ldots,d\ \}, \quad (12)$$

where x_L^i and x_H^i are the low and high boundaries of the element in the ith dimension. We also define the surface elements $\tilde{\boldsymbol{K}}^i = \times_{j\neq i}^d K^j$ (so that $\boldsymbol{K} = \tilde{\boldsymbol{K}}^i \times K^i$), then set $\boldsymbol{x} = \{x^i, \tilde{\boldsymbol{x}}^i\}$ to distinguish coordinates parallel and perpendicular to the ith dimension, and the element width $\Delta x^i = (x_H^i - x_L^i)$ and center $x_C^i = \frac{1}{2}(x_L^i + x_H^i)$. We also define $|\boldsymbol{K}| = \prod_{i=1}^d \Delta x^i$ and $|\tilde{\boldsymbol{K}}^i| = \prod_{j=1,j\neq i}^d \Delta x^j$. We let the volume of an element be denoted

$$V_{\boldsymbol{K}} = \int_{\boldsymbol{K}} dV_h, \quad \text{where} \quad dV_h = \sqrt{\gamma_h}\prod_{i=1}^d dx^i, \quad (13)$$

where γ_h is the determinant of the approximate spatial metric $(\gamma_h)_{ij}$. We will discuss the approximation to the spatial metric in more detail below.

On each element, we define the approximation space consisting of functions ψ_h

$$\mathbb{V}_h^k = \{\psi_h: \psi_h|_{\boldsymbol{K}} \in \mathbb{Q}^k(\boldsymbol{K}), \forall \boldsymbol{K} \in \mathcal{T}\}, \quad (14)$$

where $\mathbb{Q}^k$ is the tensor product space of 1D polynomials of maximal degree k. In the DG method, the functions in $\mathbb{V}_h^k$ can be discontinuous across element interfaces. In `thornado`, we use Lagrange polynomials,

$$\ell_p(\xi^i) = \prod_{\substack{q=1\\q\neq p}}^N \frac{\xi^i - \xi_q^i}{\xi_p^i - \xi_q^i}, \quad (15)$$

where $N = k + 1$ and the polynomials ℓ_p are defined on the unit reference interval $I^i = \left\{\xi^i: \xi^i \in \left(-\frac{1}{2}, \frac{1}{2}\right)\right\}$ $(i = 1,\ldots, d)$. The physical coordinate x^i is related to the reference coordinate ξ^i by the transformation $x^i(\xi^i) = x_C^i + \Delta x^i\,\xi^i$. For the Lagrange polynomials, we define the set of interpolation points $S_N^i = \{\xi_1^i,\ldots,\xi_N^i\} \subseteq I^i$. Note that for $\xi_q^i \in S_N^i$, we have $\ell_p(\xi_q^i) = \delta_{pq}$, where δ_{pq} is the Kronecker delta. As an example, the multidimensional basis function $\phi_{\boldsymbol{i}}(\boldsymbol{x}(\boldsymbol{\xi})) \in \mathbb{V}_h^k$ takes the form

$$\phi_{\boldsymbol{i}}(\boldsymbol{\xi}) = \phi_{\{i_1,\ldots,i_d\}}(\xi^1,\ldots,\xi^d) = \ell_{i_1}(\xi^1)\times\ldots\times\ell_{i_d}(\xi^d), \quad (16)$$

where we have introduced the multi-index $\boldsymbol{i} = \{i_1,\ldots,i_d\} \in \mathbb{N}^d$ (a d-tuple) to achieve a more compact notation. To further illustrate, in each element $\boldsymbol{K}$, we approximate the solution to Equation (7) by $\boldsymbol{U}_h$, which is given by an expansion of functions in $\mathbb{V}_h^k$ of the form

$$\boldsymbol{U}_h(\boldsymbol{x}, t) = \sum_{\boldsymbol{i}=\boldsymbol{1}}^{\boldsymbol{N}} \boldsymbol{U}_{\boldsymbol{i}}(t)\phi_{\boldsymbol{i}}(\boldsymbol{x}(\boldsymbol{\xi})) = \sum_{i_1=1}^N \ldots \sum_{i_d=1}^N \times \boldsymbol{U}_{\{i_1,\ldots,i_d\}}(t)\ell_{i_1}(\xi^1)\times\ldots\times\ell_{i_d}(\xi^d), \quad (17)$$

where $\boldsymbol{N} \in \mathbb{N}^d$ is the d-tuple $\{N,\ldots,N\}$. The DG method does not require the approximate multidimensional solution to be constructed from 1D polynomials of the same degree k in each dimension, but we make this choice. In the multidimensional setting, we denote the set of interpolation points in element $\boldsymbol{K}$ by $\boldsymbol{S}_N = \otimes_{i=1}^d S_N^i$. For $\boldsymbol{\xi_j} \in \boldsymbol{S}_N$, we have $\phi_{\boldsymbol{i}}(\boldsymbol{\xi_j}) = \delta_{\boldsymbol{ij}} = \delta_{i_1 j_1}\times\ldots\times\delta_{i_d j_d}$, which follows from the Kronecker delta property of the Lagrange polynomials emphasized above. Therefore, for $\boldsymbol{\xi_j} \in \boldsymbol{S}_N$, a direct evaluation in Equation (17) shows that $\boldsymbol{U}_h(\boldsymbol{x}(\boldsymbol{\xi_j})) = \boldsymbol{U}_{\boldsymbol{j}}(t)$; i.e., the expansion coefficients in Equation (17)—the unknowns to be determined by the DG method—are simply the evolved quantities evaluated in the interpolation points on each element.

We are now ready to state the DG formulation, which forms the basis for the DG method implemented in `thornado`. The semidiscrete DG problem is to find $\boldsymbol{U}_h \in \mathbb{V}_h^k$, which approximates $\boldsymbol{U}$ in Equation (7), such that

$$\langle \partial_t \boldsymbol{U}_h, \psi_h\rangle_{\boldsymbol{K}} = \mathcal{B}_h^{\mathsf{Flx}}(\boldsymbol{U}_h, \psi_h)_{\boldsymbol{K}} + \langle \boldsymbol{S}(\boldsymbol{U}_h, \Phi_h), \psi_h\rangle_{\boldsymbol{K}} \equiv \mathcal{B}_h(\boldsymbol{U}_h, \Phi_h, \psi_h)_{\boldsymbol{K}} \quad (18)$$

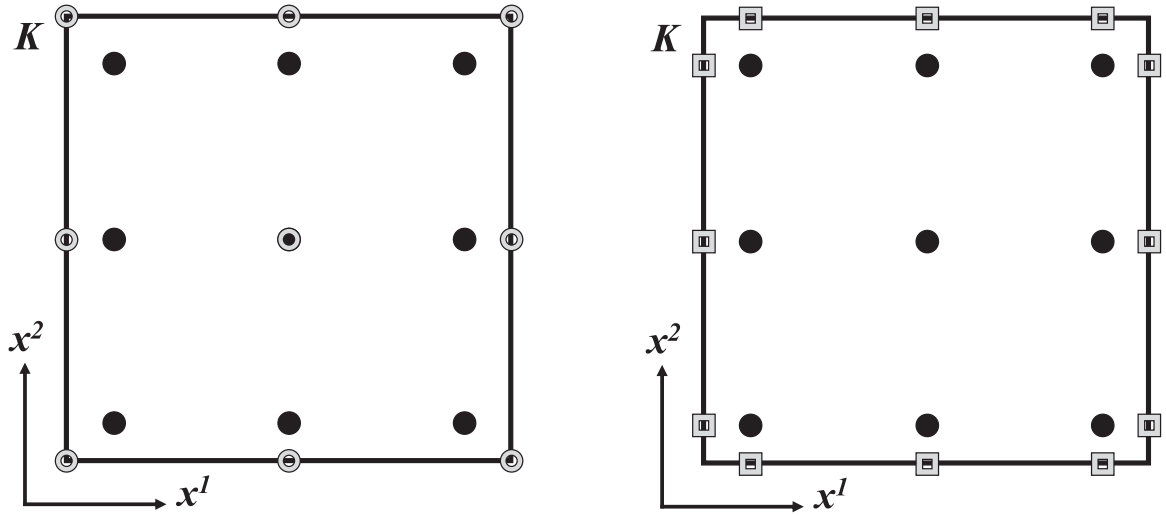

Figure 1. Reference elements with interpolation and quadrature points used in the DG method implemented in `thornado` for the 2D case ($d = 2$) with polynomials of degree $k = 2$ ($N = 3$). In the left panel, interpolation points are shown for the hydrodynamics variables ($\boldsymbol{S}_N$ (based on LG quadrature points; black, filled circles)) and the geometry scale factors and the Newtonian gravitational potential ($\hat{\boldsymbol{S}}_N$ (based on LGL quadrature points; gray, open circles)). In the right panel, quadrature points associated with volume integrals (black, filled circles) and surface integrals (gray, open squares) are shown. Note that in the collocation nodal DG method, the interpolation points in the left panel, $\boldsymbol{S}_N$, coincide with the quadrature points in the right panel. The quadrature points on the surface of the element are obtained as the projection of the quadrature points inside the element onto each surface.

holds for all test functions $\psi_h \in \mathbb{V}_h^k$ and all elements $\boldsymbol{K} \in \mathcal{T}$. In Equation (18),

$$\langle\, \partial_t \boldsymbol{U}_h,\, \psi_h \,\rangle_{\boldsymbol{K}} = \int_{\boldsymbol{K}} \partial_t \boldsymbol{U}_h\, \psi_h\, dV_h, \tag{19}$$

and we have defined the contributions from the fluxes as

$$\mathcal{B}_h^{\mathsf{Flx}}(\boldsymbol{U}_h,\, \psi_h)_{\boldsymbol{K}} = -\sum_{i=1}^{d} \int_{\tilde{\boldsymbol{K}}^i} (\sqrt{\gamma_h}\, \widehat{\boldsymbol{F}}^i(\boldsymbol{U}_h)\psi_h|_{x_{\mathrm{H}}^i} - \sqrt{\gamma_h}\, \widehat{\boldsymbol{F}}^i(\boldsymbol{U}_h)\psi_h|_{x_{\mathrm{L}}^i}) d\tilde{\boldsymbol{x}}^i + \sum_{i=1}^{d} \int_{\boldsymbol{K}} \boldsymbol{F}^i(\boldsymbol{U}_h)\partial_i\psi_h\, dV_h, \tag{20}$$

and the contributions from the sources as

$$\langle\, \boldsymbol{S}(\boldsymbol{U}_h,\, \Phi_h),\, \psi_h \,\rangle_{\boldsymbol{K}} = \int_{\boldsymbol{K}} \boldsymbol{S}(\boldsymbol{U}_h,\, \Phi_h)\psi_h\, dV_h. \tag{21}$$

The approximation to the Newtonian gravitational potential, denoted Φ_h (not to be confused with the basis functions $\phi_{\boldsymbol{i}}$ in Equation (17)), is obtained by solving Equation (5) using a finite-element method. We discuss this in Section 3.5.

In Equation (20), the numerical flux $\widehat{\boldsymbol{F}}^i(\boldsymbol{U}_h)$ is introduced to define a unique flux in the ith surface of $\boldsymbol{K}$. This numerical flux is computed from a numerical flux function (obtained, e.g., from solving an approximate Riemann problem),

$$\widehat{\boldsymbol{F}}^i(\boldsymbol{U}_h;\, x^i,\, \tilde{\boldsymbol{x}}^i) = \boldsymbol{f}^i(\boldsymbol{U}_h(x^{i,-},\, \tilde{\boldsymbol{x}}^i),\, \boldsymbol{U}_h(x^{i,+},\, \tilde{\boldsymbol{x}}^i)), \tag{22}$$

where the superscripts $-/+$ in the arguments of $\boldsymbol{U}_h(x^{i,-/+},\, \tilde{\boldsymbol{x}}^i)$ indicate that the approximation is evaluated to the immediate left/right of the interface located at x^i. In `thornado`, we have implemented the Harten–Lax–van Leer (HLL; Harten et al. 1983) and HLL-Contact (HLLC; Toro et al. 1994) flux functions, but in the numerical experiments in Sections 4 and 5, we use exclusively the HLL flux function given by

$$\boldsymbol{f}^i(\boldsymbol{U}_h^-,\, \boldsymbol{U}_h^+) = \frac{\alpha^{i,+}\, \boldsymbol{F}^i(\boldsymbol{U}_h^-) + \alpha^{i,-}\, \boldsymbol{F}^i(\boldsymbol{U}_h^+) - \alpha^{i,-}\alpha^{i,+}(\boldsymbol{U}_h^+ - \boldsymbol{U}_h^-)}{\alpha^{i,-} + \alpha^{i,+}}, \tag{23}$$

where $\boldsymbol{U}_h^{\pm} = \boldsymbol{U}_h(x^{i,\pm},\, \tilde{\boldsymbol{x}}^i)$, and where $\alpha^{i,-}$ and $\alpha^{i,+}$ are wave speed estimates for the fastest (in absolute value; $\alpha^{i,\pm} \geqslant 0$) left- and right-propagating waves, respectively. For these estimates, we simply use (Davies 1988)

$$\alpha^{i,-} = \max_{j\in\{1,\ldots,6\}}(0,\, -\lambda_j^i(\boldsymbol{U}_h^-),\, -\lambda_j^i(\boldsymbol{U}_h^+)) \quad \text{and}$$
$$\alpha^{i,+} = \max_{j\in\{1,\ldots,6\}}(0,\, +\lambda_j^i(\boldsymbol{U}_h^-),\, +\lambda_j^i(\boldsymbol{U}_h^+)), \tag{24}$$

where λ_j^i are the eigenvalues of the flux Jacobian introduced in Equation (10).

Motivated by results presented by Bassi et al. (2013), we employ a spectral-type collocation nodal DG method in `thornado`. To this end, we use Legendre–Gauss (LG) points to construct the interpolation points comprising $\boldsymbol{S}_N$. See the left panel of Figure 1 for the distribution of the interpolation points $\boldsymbol{S}_N$ in the 2D case with $k = 2$ (black, filled circles). In the collocation nodal DG method, these interpolation points are also used as quadrature points to evaluate integrals in Equation (18). One of the benefits of this collocation method is computational efficiency because, even when using curvilinear coordinates, the mass matrix associated with the term in Equation (19) is diagonal and easily invertible. On the other hand, demanding exact evaluation of integrals—e.g., by using an extended quadrature set—results in mass matrices that are nondiagonal and vary from element to element because of the spatially dependent metric determinant in dV_h in Equation (19). The use of LG points, as opposed to Legendre–Gauss–Lobatto (LGL) points, provides better accuracy in evaluating the integrals. In the 1D setting, the N-point LG quadrature evaluates polynomials of degree up to $2N - 1$ exactly, while

the corresponding LGL quadrature evaluates polynomials of degree up to $2N-3$ exactly. Let Q_N^i denote the 1D N-point LG quadrature on the interval I^i with abscissas $\{\xi_q^i\}_{q=1}^N$ and weights $\{w_q^i\}_{q=1}^N$, normalized so that $\sum_{q=1}^N w_q^i = 1$. (Note that quadrature points and weights defined on the commonly used reference interval $[-1, 1]$ (e.g., Cockburn & Shu 2001) must be scaled by a factor of $1/2$ before use on the reference interval $[-1/2, 1/2]$ used in `thornado`.) Multidimensional integrals are evaluated by the tensorization of 1D quadratures. For volume integrals over the multidimensional reference element $\boldsymbol{I} = \times_{i=1}^d I^i$, we let $\boldsymbol{Q}_N = \otimes_{i=1}^d Q_N^i$ denote the tensorization of 1D N-point LG quadrature rules with abscissas $\{\boldsymbol{\xi_q}\}_{\boldsymbol{q}=\mathbf{1}}^{\boldsymbol{N}}$ and weights $\{w_{\boldsymbol{q}}\}_{\boldsymbol{q}=\mathbf{1}}^{\boldsymbol{N}}$, where $\boldsymbol{q} = \{q_1,\ldots,q_d\} \in \mathbb{N}^d$, $\boldsymbol{\xi_q} = \{\xi_{q_1}^1,\ldots,\xi_{q_d}^d\}$, and $w_{\boldsymbol{q}} = w_{q_1}\times \ldots \times w_{q_d}$, so that the integral of a polynomial $P(\boldsymbol{x}) \in \mathbb{V}_h^k$ in element $\boldsymbol{K}$ is evaluated as

$$\begin{aligned}\int_{\boldsymbol{K}} P(\boldsymbol{x})d\boldsymbol{x} &= |\boldsymbol{K}|\int_{\boldsymbol{I}} P(\boldsymbol{\xi})d\boldsymbol{\xi} = |\boldsymbol{K}|\,\boldsymbol{Q}_N[P(\boldsymbol{\xi})]\\ &= |\boldsymbol{K}|\sum_{\boldsymbol{q}=\mathbf{1}}^{\boldsymbol{N}} w_{\boldsymbol{q}}\,P(\boldsymbol{\xi_q})\\ &= \Delta x^1\times \ldots \times \Delta x^d \sum_{q_1=1}^N \ldots \sum_{q_d=1}^N w_{q_1}\\ &\quad \times \ldots \times w_{q_d}\,P(\xi_{q_1}^1,\ldots,\xi_{q_d}^d).\end{aligned} \tag{25}$$

Similarly, for surface integrals over the reference surface element $\tilde{\boldsymbol{I}}^i = \times_{j=1,j\neq i}^d I^j$, we let $\tilde{\boldsymbol{Q}}_N^i = \otimes_{j=1,j\neq i}^d Q_N^j$ denote the tensorization of 1D N-point LG quadrature rules with abscissas $\{\tilde{\boldsymbol{\xi}}_{\tilde{\boldsymbol{q}}_i}^{\,i}\}_{\tilde{\boldsymbol{q}}_i=\mathbf{1}}^{\boldsymbol{N}}$ and weights $\{w_{\tilde{\boldsymbol{q}}_i}\}_{\tilde{\boldsymbol{q}}_i=\mathbf{1}}^{\boldsymbol{N}}$, where $\tilde{\boldsymbol{q}}_i = \{q_j\}_{j=1,j\neq i}^d \in \mathbb{N}^{d-1}$, $\tilde{\boldsymbol{\xi}}_{\tilde{\boldsymbol{q}}_i}^{\,i} = \{\xi_{q_j}^j\}_{j=1,j\neq i}^d$, and $w_{\tilde{\boldsymbol{q}}_i} = \prod_{j=1,j\neq i}^d w_{q_j}$, so that for $P(x^i, \tilde{\boldsymbol{x}}^i) \in \mathbb{V}_h^k$, the integral over the surface element $\tilde{\boldsymbol{K}}^i$ is evaluated as

$$\begin{aligned}\int_{\tilde{\boldsymbol{K}}^i} P(x^i, \tilde{\boldsymbol{x}}^i)d\tilde{\boldsymbol{x}}^i &= |\tilde{\boldsymbol{K}}^i|\int_{\tilde{\boldsymbol{I}}^i} P(x^i, \tilde{\boldsymbol{\xi}}^i)d\tilde{\boldsymbol{\xi}}^i\\ &= |\tilde{\boldsymbol{K}}^i|\,\tilde{\boldsymbol{Q}}_N^{\,i}[P(x^i, \tilde{\boldsymbol{\xi}}^i)]\\ &= |\tilde{\boldsymbol{K}}^i|\sum_{\tilde{\boldsymbol{q}}_i=\mathbf{1}}^{\boldsymbol{N}} w_{\tilde{\boldsymbol{q}}_i}\,P(x^i, \tilde{\boldsymbol{\xi}}_{\tilde{\boldsymbol{q}}_i}^{\,i})\\ &\overset{(i=1)}{=} \Delta x^2\times \ldots \times \Delta x^d \sum_{q_2=1}^N \ldots \sum_{q_d=1}^N w_{q_2}\\ &\quad \times \ldots \times w_{q_d}\,P(x^1, \xi_{q_2}^2,\ldots,\xi_{q_d}^d),\end{aligned} \tag{26}$$

where the specific case with $i = 1$ is given in the second line. The points used to evaluate volume integrals with the $\boldsymbol{Q}_N$ quadrature rule for the case with $d = k = 2$ are shown as black, filled circles in the right panel in Figure 1. (Note that these points are identical to the interpolation points displayed as black, filled circles in the left panel in Figure 1.) The quadrature points used to evaluate surface integrals with $\tilde{\boldsymbol{Q}}^1$ and $\tilde{\boldsymbol{Q}}^2$ are shown as the gray, open squares on the boundary of the element.

By inserting the expansion in Equation (17), letting $\psi_h = \phi_{\boldsymbol{p}}$, where $\phi_{\boldsymbol{p}}$ is one of the basis functions in the expansion in Equation (17), and using the quadrature rule in Equation (25), we can evaluate Equation (19) as

$$\langle\, \partial_t \boldsymbol{U}_h,\, \phi_{\boldsymbol{p}}\,\rangle_{\boldsymbol{K}} := w_{\boldsymbol{p}}\,|\boldsymbol{K}|\,\sqrt{\gamma_{\boldsymbol{p}}}\;\partial_t \boldsymbol{U}_{\boldsymbol{p}}, \tag{27}$$

where $w_{\boldsymbol{j}}\,|\boldsymbol{K}|\,\sqrt{\gamma_{\boldsymbol{j}}}$ are the elements of the diagonal mass matrix and $\gamma_{\boldsymbol{j}} = \gamma_h(\boldsymbol{x_j})$. Similarly, using the quadrature in Equation (26), the contributions from fluxes can be written as

$$\begin{aligned}\mathcal{B}_h^{\mathsf{Flx}}(\boldsymbol{U}_h,\, \phi_{\boldsymbol{p}})_{\boldsymbol{K}} := -\sum_{i=1}^d w_{\tilde{\boldsymbol{p}}_i}\,|\tilde{\boldsymbol{K}}^i|\big(&\sqrt{\gamma_h(x_{\mathrm{H}}^i, \tilde{\boldsymbol{x}}_{\tilde{\boldsymbol{p}}_i}^i)}\\ &\times \widehat{\boldsymbol{F}}^i(x_{\mathrm{H}}^i, \tilde{\boldsymbol{x}}_{\tilde{\boldsymbol{p}}_i}^i)\ell_{p_i}(x_{\mathrm{H}}^{i,-})\\ &- \sqrt{\gamma_h(x_{\mathrm{L}}^i, \tilde{\boldsymbol{x}}_{\tilde{\boldsymbol{p}}_i}^i)}\;\widehat{\boldsymbol{F}}^i(x_{\mathrm{L}}^i, \tilde{\boldsymbol{x}}_{\tilde{\boldsymbol{p}}_i}^i)\ell_{p_i}(x_{\mathrm{L}}^{i,+})\big)\\ &+ \sum_{i=1}^d w_{\tilde{\boldsymbol{p}}_i}\,|\tilde{\boldsymbol{K}}^i|\sum_{q_i=1}^N w_{q_i}\sqrt{\gamma_h(x_{q_i}^i, \tilde{\boldsymbol{x}}_{\tilde{\boldsymbol{p}}_i}^i)}\;\boldsymbol{F}^i(x_{q_i}^i, \tilde{\boldsymbol{x}}_{\tilde{\boldsymbol{p}}_i}^i)\frac{\partial \ell_{p_i}}{\partial \xi^i}(\xi_{q_i}^i).\end{aligned} \tag{28}$$

Finally, the source term becomes

$$\langle\, \boldsymbol{S}(\boldsymbol{U}_h, \Phi_h),\, \phi_{\boldsymbol{p}}\,\rangle_{\boldsymbol{K}} := w_{\boldsymbol{j}}\,|\boldsymbol{K}|\,\sqrt{\gamma_{\boldsymbol{p}}}\;\boldsymbol{S}_{\boldsymbol{p}}, \tag{29}$$

where $\boldsymbol{S}_{\boldsymbol{p}}$ is the source vector in Equation (8), evaluated in $\boldsymbol{x}_{\boldsymbol{p}}$. Combining Equations (27)–(29), we can now write the spectral-type collocation DG approximation to the semidiscrete DG problem in Equation (18) in terms of an evolution equation for the expansion coefficient $\boldsymbol{U}_{\boldsymbol{p}}$ in element $\boldsymbol{K}$ as

$$\begin{aligned}\partial_t \boldsymbol{U}_{\boldsymbol{p}} = -\sum_{i=1}^d \frac{1}{w_{p_i}\Delta x^i\,\sqrt{\gamma_{\boldsymbol{p}}}}\big(&\sqrt{\gamma_h(x_{\mathrm{H}}^i, \tilde{\boldsymbol{x}}_{\tilde{\boldsymbol{p}}_i}^i)}\;\widehat{\boldsymbol{F}}^i(x_{\mathrm{H}}^i, \tilde{\boldsymbol{x}}_{\tilde{\boldsymbol{p}}_i}^i)\ell_{p_i}(x_{\mathrm{H}}^{i,-})\\ &- \sqrt{\gamma_h(x_{\mathrm{L}}^i, \tilde{\boldsymbol{x}}_{\tilde{\boldsymbol{p}}_i}^i)}\;\widehat{\boldsymbol{F}}^i(x_{\mathrm{L}}^i, \tilde{\boldsymbol{x}}_{\tilde{\boldsymbol{p}}_i}^i)\ell_{p_i}(x_{\mathrm{L}}^{i,+})\\ &+ \sum_{i=1}^d \frac{1}{w_{p_i}\Delta x^i\,\sqrt{\gamma_{\boldsymbol{p}}}}\sum_{q_i=1}^N w_{q_i}\sqrt{\gamma_h(x_{q_i}^i, \tilde{\boldsymbol{x}}_{\tilde{\boldsymbol{p}}_i}^i)}\\ &\times \boldsymbol{F}^i(x_{q_i}^i, \tilde{\boldsymbol{x}}_{\tilde{\boldsymbol{p}}_i}^i)\frac{\partial \ell_{p_i}}{\partial \xi^i}(\xi_{q_i}^i) + \boldsymbol{S}_{\boldsymbol{p}}.\end{aligned} \tag{30}$$

(For an example of Equation (30) in the simpler 1D setting, see Endeve et al. 2019, their Equation (11).)

The cell averages in element $\boldsymbol{K}$, defined as

$$\begin{aligned}&\boldsymbol{U}_{\boldsymbol{K}} = \frac{1}{V_{\boldsymbol{K}}}\int_{\boldsymbol{K}} \boldsymbol{U}_h\,dV_h := \frac{\sum_{\boldsymbol{p}=\mathbf{1}}^{\boldsymbol{N}} w_{\boldsymbol{p}}\,\sqrt{\gamma_{\boldsymbol{p}}}\;\boldsymbol{U}_{\boldsymbol{p}}}{\sum_{\boldsymbol{p}=\mathbf{1}}^{\boldsymbol{N}} w_{\boldsymbol{p}}\,\sqrt{\gamma_{\boldsymbol{p}}}}, \quad \text{where}\\ &V_{\boldsymbol{K}} = |\boldsymbol{K}|\sum_{\boldsymbol{p}=\mathbf{1}}^{\boldsymbol{N}} w_{\boldsymbol{p}}\,\sqrt{\gamma_{\boldsymbol{p}}},\end{aligned} \tag{31}$$

play an important role in the analysis and implementation of the DG method given by Equation (30). (Examples of the use of the cell averages are given in Sections 3.3 and 3.4, where we discuss limiting techniques.) From the definition of the cell average in Equation (31) and from Equation (30), the equation for the cell average can be written as

$$\begin{aligned}\partial_t \boldsymbol{U}_{\boldsymbol{K}} = -\frac{|\boldsymbol{K}|}{V_{\boldsymbol{K}}}\sum_{i=1}^d \tilde{\boldsymbol{Q}}_N^i\big[\,&\sqrt{\gamma_h}(x_{\mathrm{H}}^i, \tilde{\boldsymbol{x}}^i)\widehat{\boldsymbol{F}}^i(x_{\mathrm{H}}^i, \tilde{\boldsymbol{x}}^i)\\ &- \sqrt{\gamma_h}(x_{\mathrm{L}}^i, \tilde{\boldsymbol{x}}^i)\widehat{\boldsymbol{F}}^i(x_{\mathrm{L}}^i, \tilde{\boldsymbol{x}}^i)\big]/\Delta x^i + \boldsymbol{S}_{\boldsymbol{K}},\end{aligned} \tag{32}$$

where we used the quadrature rule in Equation (26) to represent the surface integrals, while the source term can be written in

terms of the quadrature rule in Equation (25)

$$\boldsymbol{S}_{\boldsymbol{K}} = \frac{|\boldsymbol{K}|}{V_{\boldsymbol{K}}}\, \boldsymbol{Q}_N[\, \sqrt{\gamma_h}(\boldsymbol{x})\boldsymbol{S}(\boldsymbol{x})]. \tag{33}$$

To arrive at Equation (32), we used the property of the Lagrange polynomial in Equation (15) that $\sum_{p=1}^{N} \ell_p(\xi^i) = 1$ for any $\xi^i \in I^i$. Equation (32) exhibits the expected conservation form, with quadrature rules replacing integrals over the surface of $\boldsymbol{K}$. In the absence of sources, the DG discretization in Equation (30) is conservative for mass, momentum, energy, and electron number. We also note that Equation (32) is familiar from the literature on finite-volume (FV) methods, which only evolve the cell averages. The DG and FV methods are in fact equivalent in the first-order case, when $k = 0$. However, for the extension to higher orders, FV methods reconstruct a local polynomial using cell averages in neighboring elements, while DG methods evolve all the degrees of freedom in the local polynomial representation, so that the reconstruction step is not needed. Thus, one benefit of avoiding the reconstruction step becomes clear in the high-order case: while the FV stencil width increases with increasing spatial order of accuracy, the DG method only requires data from the local element and its nearest neighbors, independent of the order of accuracy.

We complete the specification of the basic DG method implemented in `thornado` by discussing the source terms due to the use of curvilinear coordinates and gravitational fields. In particular, we write (see Equation (9))

$$\boldsymbol{S}_{\boldsymbol{p}} = \boldsymbol{S}_{\boldsymbol{p}}^{\gamma} + \boldsymbol{S}_{\boldsymbol{p}}^{\Phi}. \tag{34}$$

3.1.1. Geometric Source Terms

For the sources due to curvilinear coordinates, $\boldsymbol{S}_{\boldsymbol{p}}^{\gamma}$, the only nonzero components appear in the components of the momentum equation, which can be written in terms of the scale factors where, due to the diagonal metric, $\gamma_{ii} = h_i h_i$ and $\gamma^{ii} = 1/\gamma_{ii}$,

$$\begin{aligned}\frac{1}{2}\,\Pi^{ik}\partial_j\gamma_{ik} &= \frac{1}{2}\,\Pi^{11}\partial_j\gamma_{11} + \frac{1}{2}\,\Pi^{22}\partial_j\gamma_{22} + \frac{1}{2}\,\Pi^{33}\partial_j\gamma_{33} \\ &= \Pi^{1}{}_{1}\frac{1}{h_1}\frac{\partial h_1}{\partial x^j} + \Pi^{2}{}_{2}\frac{1}{h_2}\frac{\partial h_2}{\partial x^j} + \Pi^{3}{}_{3}\frac{1}{h_3}\frac{\partial h_3}{\partial x^j}.\end{aligned} \tag{35}$$

For the coordinate systems we consider here, the scale factors are independent of x^3, and only the first and second components of Equation (35) are nonzero (i.e., $j = 1, 2$; see Table 1). Note that $h_1 = 1$ for all the coordinate systems; therefore, spatial derivatives of h_1 vanish. For Cartesian coordinates, the scale factors are unity, and all the components of $\boldsymbol{S}_{\boldsymbol{p}}^{\gamma}$ vanish. For cylindrical coordinates, only $h_3 = R$ contributes, while for spherical-polar coordinates both $h_2 = r$ and $h_3 = r\sin\theta$ contribute. In `thornado`, we approximate the scale factors by polynomials in each element. To this end, we define $\boldsymbol{h} = (h_1, h_2, h_3)^{\mathrm{T}}$ and let the scale factors in $\boldsymbol{K}$ be given by the expansion

$$\boldsymbol{h}_h(\boldsymbol{x}) = \sum_{\boldsymbol{i}=\boldsymbol{1}}^{\boldsymbol{N}} \boldsymbol{h_i}\, \hat{\phi}_{\boldsymbol{i}}(\boldsymbol{x}) \in \mathbb{V}_h^k, \tag{36}$$

where $\hat{\phi}_{\boldsymbol{i}}(\boldsymbol{x})$ are basis functions, similar to those defined in Equation (16). However, we demand that the scale factors be continuous across element interfaces. To achieve this, we let $\hat{S}_N^i = \{\hat{\xi}_1, \ldots, \hat{\xi}_{S_N}^i\} \subseteq I^i$ denote the set of LGL points in the unit reference interval, because the LGL points include the end points of I^i. For the scale factors (and, as discussed below, the Newtonian gravitational potential), we then let the interpolation points on $\boldsymbol{K}$ be given by $\hat{\boldsymbol{S}}_N = \otimes_{i=1}^{d} \hat{S}_N^i$. The distribution of the interpolation points $\hat{\boldsymbol{S}}_N$, used for the scale factors and the Newtonian gravitational potential, for the 2D case with $k = 2$ are shown in the left panel of Figure 1 (gray, open circles). Hence, $\hat{\phi}_{\boldsymbol{i}}(\boldsymbol{x})$ is defined as in Equation (16), but with the Lagrange polynomials in Equation (15) constructed with the LGL points $\hat{\boldsymbol{S}}_N$, and the expansion coefficients $\boldsymbol{h_i}$ are given by the exact value of the scale factors in the LGL points. Scale factors in the LG points $\boldsymbol{x_i} = \boldsymbol{S}_N$, which are needed, e.g., to compute the determinant of the spatial metric, are obtained from the direct evaluation of Equation (36), $\boldsymbol{h}_h(\boldsymbol{x_i})$, so that $\gamma_{\boldsymbol{i}} = \gamma_h(\boldsymbol{x_i}) := \gamma(\boldsymbol{h}_h(\boldsymbol{x_i}))$. Derivatives of the scale factors, needed for the source terms in Equation (35), are evaluated by analytic differentiation of Equation (36). Because in the present case the metric is time independent, the needed scale factors and their derivatives can be precomputed at program startup and stored for later use. Note that scale factors are polynomials and at most linear functions of the spherical-polar or cylindrical radius, so the representation is exact in the x^1 dimension if $N \geqslant 2$. However, for spherical-polar coordinates, h_3 is a trigonometric function in the x^2 dimension, and the representation in Equation (36) is only approximate.

Next, we consider a special case where the geometric source terms, $\frac{1}{2}\Pi^{ik}\partial_j\gamma_{ik}$, and the divergence of the stress tensor, $\frac{1}{\sqrt{\gamma}}\partial_i(\sqrt{\gamma}\ \Pi^i{}_j)$, appearing in the components of the momentum equation, Equation (2), must balance each other. Specifically, for a fluid associated with an isotropic and spatially homogeneous stress tensor, i.e., $\Pi^i{}_k = p_0\ \delta^i{}_k$ (p_0 = constant), the divergence of the stress tensor must balance the geometry source exactly to prevent inducing spurious flows.

Considering Equation (32), with Equations (33) and (35), in spherical-polar coordinates and in the absence of gravity, assuming an isotropic and spatially homogeneous stress tensor, the equation for the first component of the momentum density (see Equation (8)), in the sense of the cell average, can be written as

$$\begin{aligned}\partial_t(\rho v_1)_{\boldsymbol{K}} &= -p_0\,\frac{|\tilde{\boldsymbol{K}}^1|}{V_{\boldsymbol{K}}}\{\tilde{\boldsymbol{Q}}_N^1[\,\sqrt{\gamma_h}(x_{\mathrm{H}}^1, \tilde{\boldsymbol{x}}^1) \\ &\quad - \sqrt{\gamma_h}(x_{\mathrm{L}}^1, \tilde{\boldsymbol{x}}^1)] - \boldsymbol{Q}_N[\,2\sqrt{\gamma_h}(\boldsymbol{x})/x^1\,]\} \\ &= -p_0\,\frac{|\tilde{\boldsymbol{K}}^1|}{V_{\boldsymbol{K}}}\,\tilde{\boldsymbol{Q}}_N^1[(\sin\theta)_h((r_{\mathrm{H}}^2 - r_{\mathrm{L}}^2) - 2\,Q_N^1[r])],\end{aligned} \tag{37}$$

where $(\sin\theta)_h$ is the polynomial approximation to $\sin\theta$. Because the stress tensor is isotropic and spatially homogeneous, the numerical flux in the first component of the momentum equation is simply $\widehat{F}^i_{(\rho v_1)} = p_0\,\delta^i{}_1$. The right-hand side of Equation (37) vanishes because the LG quadrature, with $N \geqslant 1$, is exact for the radial integral; i.e., $2\,Q_N^1[r] = (r_{\mathrm{H}}^2 - r_{\mathrm{L}}^2)$. Similarly, the second component of the momentum equation

can be written as

$$\partial_t(\rho v_2)_{\boldsymbol{K}} = -p_0 \frac{|\tilde{\boldsymbol{K}}^2|}{V_{\boldsymbol{K}}} \tilde{\boldsymbol{Q}}_N^2 [\, r^2(\sin\theta_{\rm H} - \sin\theta_{\rm L}) - Q_N^2[\partial_{\xi^2}(\sin\theta)_h])]. \quad (38)$$

Because $(\sin\theta)_h$ is approximated by a polynomial of degree $k = N-1$, the N-point LG quadrature in the θ direction is evaluated exactly, so that $Q_N^2[\partial_{\xi^2}(\sin\theta)_h] = (\sin\theta_{\rm H} - \sin\theta_{\rm L})$, which implies that the right-hand side of Equation (38) vanishes. Note that these properties hold for polynomial approximations with $k \geqslant 1$. The first-order accurate scheme ($k = 0$) requires special treatment and is not discussed here. (See, e.g., Mönchmeyer & Müller 1989 and Blondin & Lufkin 1993 for finite-volume schemes and associated challenges when using spherical-polar coordinates.)

In cylindrical coordinates, the source term in Equation (35) contributes only to the first component of the momentum equation. In this case, the equation for the cell average can be written as

$$\partial_t(\rho v_1)_{\boldsymbol{K}} = -p_0 \frac{|\tilde{\boldsymbol{K}}^1|}{V_{\boldsymbol{K}}} \tilde{\boldsymbol{Q}}_N^1 [(R_{\rm H} - R_{\rm L}) - Q_N^1[\partial_{\xi^1} R]\,]. \quad (39)$$

Again, because the quadrature in the R direction is exact, $Q_N^1[\partial_{\xi^1} R] = (R_{\rm H} - R_{\rm L})$, and the right-hand side of Equation (39) vanishes, as is desired under the conditions of an isotropic and spatially homogeneous stress tensor.

3.1.2. Gravitational Source Terms

For the gravitational source terms appearing in the momentum and energy equations, our approach is similar to that used for the geometric sources discussed above. The gravitational potential in element $\boldsymbol{K}$ is approximated by the polynomial

$$\Phi_h(\boldsymbol{x}) = \sum_{\boldsymbol{i}=\boldsymbol{1}}^{\boldsymbol{N}} \Phi_{\boldsymbol{i}}\, \hat{\phi}_{\boldsymbol{i}}(\boldsymbol{x}), \quad (40)$$

constrained to be continuous on the element interfaces, so that

$$\Phi_h(x^{i,+}_{\rm L/H}, \tilde{\boldsymbol{x}}^i) = \Phi_h(x^{i,-}_{\rm L/H}, \tilde{\boldsymbol{x}}^i). \quad (41)$$

(Continuity of the potential on the element interfaces is guaranteed by the finite-element method in Section 3.5.) We then compute derivatives of the gravitational potential by analytic differentiation of the expansion in Equation (40), and write the momentum and energy sources in the interpolation point $\boldsymbol{x}_{\boldsymbol{p}} \in \boldsymbol{S}_N$ as

$$(S^{\Phi}_{\rho v_j})_{\boldsymbol{p}} = -\rho_{\boldsymbol{p}}(\partial_j \Phi_h)_{\boldsymbol{p}} \quad \text{and} \quad (S^{\Phi}_E)_{\boldsymbol{p}} = -\sum_{j=1}^{d} (\rho v^j)_{\boldsymbol{p}} (\partial_j \Phi_h)_{\boldsymbol{p}}, \quad (42)$$

where, $\rho_{\boldsymbol{p}}$, $(\rho v^j)_{\boldsymbol{p}}$, and $(\partial_j \Phi_h)_{\boldsymbol{p}}$ are, respectively, the mass density, momentum density, and the derivative of Equation (40), evaluated in $\boldsymbol{x}_{\boldsymbol{p}}$. We note that the source terms in Equation (42) are not well balanced, i.e., designed specifically to capture steady states (e.g., hydrostatic equilibrium), which would require special treatment (see, e.g., Käppeli & Mishra 2016; Li & Xing 2018).

3.2. Time Integration

After application of the DG spatial discretization, Equation (18) can be viewed as a system of ordinary differential equations (ODEs), which can be written as

$$\frac{d}{dt} \langle\, \boldsymbol{U}_h, \psi_h \,\rangle_{\boldsymbol{K}} = \mathcal{B}_h(\boldsymbol{U}_h, \Phi_h, \psi_h)_{\boldsymbol{K}}. \quad (43)$$

This system of ODEs is evolved with the explicit strong stability-preserving Runge–Kutta (SSP-RK) methods of Shu & Osher (1988; see also Cockburn & Shu 2001; Gottlieb et al. 2001). Denoting the fluid fields and the gravitational potential at time t^n by $\boldsymbol{U}_h^n$ and Φ_h^n, respectively, the time-stepping algorithm advancing the solution from t^n to $t^{n+1} = t^n + \Delta t^n$ with s stages is $\forall\, \psi_h \in \mathbb{V}_h^k$ and $\forall\, \boldsymbol{K} \in \mathcal{T}$,

Algorithm 1. Algorithm for SSP-RK Time Integration

1 $\langle \boldsymbol{U}_h^{(0)}, \psi_h \rangle_{\boldsymbol{K}} \coloneqq \Lambda^{\rm be}\{\Lambda^{\rm tvd}\{\langle \boldsymbol{U}_h^n, \psi_h \rangle_{\boldsymbol{K}}\}\}$
2 $\Phi_h^{(0)} \coloneqq \Phi_h^n$
3 for $i = 1,\ldots,s$ do
4 $\langle \boldsymbol{U}_h^{(i)}, \psi_h \rangle_{\boldsymbol{K}} \coloneqq \Lambda^{\rm be}\left\{\Lambda^{\rm tvd}\left\{\sum_{j=0}^{i-1} \alpha_{ij}\left(\langle \boldsymbol{U}_h^{(j)}, \psi_h \rangle_{\boldsymbol{K}} + \frac{\beta_{ij}}{\alpha_{ij}} \Delta t^n\, \mathcal{B}_h^{(j)}\right)\right\}\right\}$,
5 where $\mathcal{B}_h^{(j)} \coloneqq \mathcal{B}_h(\boldsymbol{U}_h^{(j)}, \Phi_h^{(j)}, \psi_h)$, with $\Phi_h^{(j)} \coloneqq \Phi(\boldsymbol{U}_h^{(j)})$
6 $\Phi_h^{(i)} \coloneqq \Phi(\boldsymbol{U}_h^{(i)})$
7 end
8 $\langle \boldsymbol{U}_h^{n+1}, \psi_h \rangle_{\boldsymbol{K}} \coloneqq \langle \boldsymbol{U}_h^{(s)}, \psi_h \rangle_{\boldsymbol{K}}$
9 $\Phi_h^{n+1} \coloneqq \Phi_h(\boldsymbol{U}_h^{n+1})$

Note that line 6 in Algorithm 1 invokes the Poisson solver for the gravitational potential. Details about the coefficients α_{ij} and β_{ij} can be found in Cockburn & Shu (2001). In order for the evolution of the cell average of the solution to be stable, the time step must satisfy the Courant–Friedrichs–Lewy (CFL) condition,

$$\Delta t \leqslant \frac{C_{\rm CFL}}{d(2k+1)} \times \min_{i \in \{1,\ldots,d\}} \left(\frac{\Delta x^i}{|\lambda^i|}\right), \quad (44)$$

where d is the number of spatial dimensions, k is the maximal degree of the 1D polynomials comprising $\mathbb{V}_h^k$, $C_{\rm CFL} \lesssim 1$ is the CFL number, and λ^i is the largest (in magnitude) eigenvalue of the flux Jacobian in Equation (10), corresponding to the fastest moving wave in the ith spatial dimension.

In principle, one would also need an additional restriction on the time step to guarantee that the solution remains in the set of physically admissible states (see Section 3.4). However, we do not enforce such a condition because in practice we find the CFL condition given by Equation (44) to be sufficient.

The operators $\Lambda^{\rm tvd}$ and $\Lambda^{\rm be}$ invoked in lines 1 and 4 in Algorithm 1 represent the slope and bound-enforcing limiters, respectively, and play an important role in RKDG methods. In particular, the slope limiter is required in order for the SSP-RK method to guarantee stability when applied to nonlinear problems (Cockburn & Shu 2001).

3.3. Slope Limiting

To improve the stability of the RKDG algorithm and prevent unphysical oscillations in the solutions around discontinuities, it is necessary to implement a limiting procedure for the polynomial $\boldsymbol{U}_h$. To this end, we use the basic minmod-type

total variation diminishing (TVD) slope limiter (see, e.g., Cockburn & Shu 1998) in conjunction with the troubled-cell indicator (TCI) proposed by Fu & Shu (2017). The TCI prevents excessive limiting by only flagging elements where limiting is needed. When using the basic TVD limiter one assumes that any spurious oscillations are evident in the part of the solution that is represented by piecewise linear functions, and under- and overshoots of the higher-order solution at intercell boundaries are detected by comparing local slopes with slopes constructed using cell averages of the target cell and its neighbors. Our implementation follows closely the description in Schaal et al. (2015) for the case of an ideal EoS. Recall from Equation (17) that in each cell the solution is expressed in the nodal form. It is convenient, however, for limiting purposes to express the solution in $\boldsymbol{K}$ using a modal representation

$$\boldsymbol{U}_h(\boldsymbol{x}, t) = \sum_{\boldsymbol{i}=\boldsymbol{1}}^{\boldsymbol{N}} \boldsymbol{C}_{\boldsymbol{i}}(t)\tilde{\phi}_{\boldsymbol{i}}(\boldsymbol{x}), \tag{45}$$

where the multidimensional modal basis functions $\tilde{\phi}_{\boldsymbol{i}}(\boldsymbol{x}(\boldsymbol{\xi})) \in \mathbb{V}_h^k$ are constructed from 1D Legendre polynomials $\{P_\ell(\xi^i)\}_{\ell=0}^{N-1}$ by tensorization, i.e.,

$$\tilde{\phi}_{\boldsymbol{i}}(\boldsymbol{\xi}) = \tilde{\phi}_{\{i_1,\ldots,i_d\}}(\xi^1,\ldots,\xi^d) = P_{i_1-1}(\xi^1)\times \ldots \times P_{i_d-1}(\xi^d). \tag{46}$$

The Legendre polynomials are orthogonal in the unit interval I^i, and in `thornado`, we use a normalization such that $P_0(\xi^i) = 1$ and $P_1(\xi^i) = \xi^i$ (i.e., the polynomials are orthogonal, but not orthonormal with respect to the standard L^2 inner product on I^i). Note that the case with $\boldsymbol{i} = \{i_1,\ldots,i_d\} = \{1,\ldots,1\} = \boldsymbol{1}$ corresponds to $\tilde{\phi}_{\boldsymbol{1}}(\boldsymbol{x}) = P_0(\xi^1)\times \ldots \times P_0(\xi^d) = 1$; therefore, the expansion coefficient $\boldsymbol{C}_{\boldsymbol{1}}$ is equal to the cell average when Cartesian coordinates are used ($\gamma_h = 1$), i.e.,

$$\boldsymbol{C}_{\boldsymbol{1}} = \frac{1}{|\boldsymbol{K}|}\int_{\boldsymbol{K}} \boldsymbol{U}_h \, d\boldsymbol{x}. \tag{47}$$

In our multi-index notation, we define $|\boldsymbol{i}| = \sum_{j=1}^{d} i_j$, so that the basis functions $\tilde{\phi}_{\boldsymbol{i}}$ with $\boldsymbol{i}$ satisfying $|\boldsymbol{i}| = d+1$ are linear in one of the coordinates. For example, for the 3D case ($d = 3$), we have exactly three basis functions satisfying $|\boldsymbol{i}| = d+1 = 4$,

$$\tilde{\phi}_{\{2,1,1\}}(\boldsymbol{x}) = P_1(\xi^1) \times P_0(\xi^2) \times P_0(\xi^3) = P_1(\xi^1) = \xi^1, \tag{48}$$

$$\tilde{\phi}_{\{1,2,1\}}(\boldsymbol{x}) = P_0(\xi^1) \times P_1(\xi^2) \times P_0(\xi^3) = P_1(\xi^2) = \xi^2, \quad \text{and} \tag{49}$$

$$\tilde{\phi}_{\{1,1,2\}}(\boldsymbol{x}) = P_0(\xi^1) \times P_0(\xi^2) \times P_1(\xi^3) = P_1(\xi^3) = \xi^3, \tag{50}$$

which are linear in the reference coordinates ξ^1, ξ^2, and ξ^3, respectively. From orthogonality of the Legendre polynomials, we can identify the expansion coefficients satisfying $|\boldsymbol{i}| = 4$ in the modal representation in Equation (45) as the average derivative of $\boldsymbol{U}_h$ with respect to the reference coordinates ξ^1, ξ^2, and ξ^3, respectively, i.e.,

$$\begin{aligned}
\boldsymbol{C}_{\{2,1,1\}} &= \frac{1}{|\boldsymbol{K}|}\int_{\boldsymbol{K}} (\partial_{\xi^1}\boldsymbol{U}_h)d\boldsymbol{x},\\
\boldsymbol{C}_{\{1,2,1\}} &= \frac{1}{|\boldsymbol{K}|}\int_{\boldsymbol{K}} (\partial_{\xi^2}\boldsymbol{U}_h)d\boldsymbol{x}, \quad \text{and}\\
\boldsymbol{C}_{\{1,1,2\}} &= \frac{1}{|\boldsymbol{K}|}\int_{\boldsymbol{K}} (\partial_{\xi^3}\boldsymbol{U}_h)d\boldsymbol{x}.
\end{aligned} \tag{51}$$

These coefficients are here obtained by taking the derivative of Equation (45) with respect to ξ^1, ξ^2, and ξ^3, respectively, and integrating over the element.

We demand the representations of the solution in Equations (17) and (45) be equivalent in the least-squares sense,

$$\begin{aligned}
&\sum_{\boldsymbol{i}=\boldsymbol{1}}^{\boldsymbol{N}} \int_{\boldsymbol{K}} (\boldsymbol{U}_{\boldsymbol{i}}(t)\phi_{\boldsymbol{i}}(\boldsymbol{x}) - \boldsymbol{C}_{\boldsymbol{i}}(t)\tilde{\phi}_{\boldsymbol{i}}(\boldsymbol{x}))\psi_h(\boldsymbol{x})d\boldsymbol{x} = 0,\\
&\forall\ \psi_h \in \mathbb{V}_h^k,
\end{aligned} \tag{52}$$

which provides a change of basis between Lagrange and Legendre polynomial representations, and relates the coefficients of nodal and modal representations by linear transformations. Setting $\psi_h = \phi_{\boldsymbol{j}}$ in Equation (52) gives the nodal coefficients in terms of the modal coefficients

$$\boldsymbol{U}_{\boldsymbol{j}} = \sum_{\boldsymbol{i}=\boldsymbol{1}}^{\boldsymbol{N}} \tilde{\phi}_{\boldsymbol{i}}(\boldsymbol{\xi}_{\boldsymbol{j}})\boldsymbol{C}_{\boldsymbol{i}}, \tag{53}$$

while setting $\psi_h = \tilde{\phi}_{\boldsymbol{j}}$ in Equation (52) gives the modal coefficients in terms of the nodal coefficients

$$\sum_{\boldsymbol{i}=\boldsymbol{1}}^{\boldsymbol{N}} \int_{\boldsymbol{I}} \tilde{\phi}_{\boldsymbol{j}}(\boldsymbol{\xi})\tilde{\phi}_{\boldsymbol{i}}(\boldsymbol{\xi})d\boldsymbol{\xi}\ \boldsymbol{C}_{\boldsymbol{i}} = \sum_{\boldsymbol{i}=\boldsymbol{1}}^{\boldsymbol{N}} \int_{\boldsymbol{I}} \tilde{\phi}_{\boldsymbol{j}}(\boldsymbol{\xi})\phi_{\boldsymbol{i}}(\boldsymbol{\xi})d\boldsymbol{\xi}\ \boldsymbol{U}_{\boldsymbol{i}}, \tag{54}$$

where the matrix on the left-hand side is diagonal and easily invertible. The matrix on the right-hand side is the same for all elements and can be precomputed at program startup and stored with minimal storage requirements. As illustrated in Equations (47) and (51), the representation in terms of Legendre polynomials $\{P_\ell\}_{\ell=0}^{N-1}$ is more convenient for limiting because the polynomial degree increases with increasing ℓ, and the identification of the expansion coefficients with average values and average derivatives is more straightforward. In the Lagrange basis, all of the basis functions have the same polynomial degree.

We perform slope limiting by comparing the weights $\boldsymbol{C}_{\boldsymbol{i}}$—which for $|\boldsymbol{i}| = d+1$ and appropriate normalization of the Legendre polynomials are equal to the first derivatives of the solution in the cell—with the limited weights $\widetilde{\boldsymbol{C}}_{\boldsymbol{i}}$, computed from

$$\begin{aligned}
&\mathcal{M}\,\widetilde{\boldsymbol{C}}_{\boldsymbol{i}} = \text{minmod}(\mathcal{M}\,\boldsymbol{C}_{\boldsymbol{i}},\ \beta_{\text{Tvd}}\,\mathcal{M}(\boldsymbol{C}_{\boldsymbol{1}}^{+} - \boldsymbol{C}_{\boldsymbol{1}}),\\
&\beta_{\text{Tvd}}\,\mathcal{M}(\boldsymbol{C}_{\boldsymbol{1}} - \boldsymbol{C}_{\boldsymbol{1}}^{-}),\\
&(\forall\,\boldsymbol{i} \text{ satisfying } |\boldsymbol{i}| = d+1),
\end{aligned} \tag{55}$$

where the multivariate minmod function is defined as

$$\begin{aligned}
&\text{minmod}(a_1, a_2, a_3)\\
&= \begin{cases} s \times \min\{|a_1|, |a_2|, |a_3|\}, & \text{if } s = \text{sign}(a_1) = \text{sign}(a_2) = \text{sign}(a_3)\\ 0, & \text{otherwise.}\end{cases}
\end{aligned} \tag{56}$$

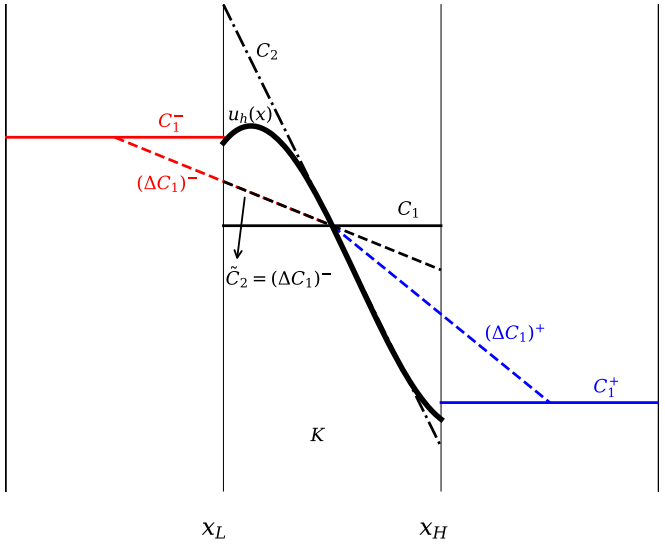

Figure 2. Illustration of how the minmod slope limiter works when applied to a 1D scalar field $U(x)$. The original, high-order polynomial $U_h(x) = \sum_{i=1}^{N} C_i\,\tilde{\phi}_i(x)$ is represented by the thick solid black curve, while its constant and linear contributions, $C_1\tilde{\phi}_1(x)$ and $C_2\tilde{\phi}_2(x)$, are represented by solid and dashed–dotted black lines, respectively. The slopes $(\Delta C_1)^+ = C_1^+ - C_1$ and $(\Delta C_1)^- = C_1 - C_1^-$—the second and third argument in the minmod function in Equation (55), respectively—are represented by the blue and red dashed lines, respectively. In this example, all three slopes have the same sign. Then, because $(\Delta C_1)^- < (\Delta C_1)^+ < C_1$, $\widetilde{C}_2 := (\Delta C_1)^-$.

The minmod function returns the minimum argument if they all have the same sign, and zero otherwise. In three spatial dimensions, we estimate limited slopes independently for all the coefficients in Equation (51), and limiting is applied to a component of $\boldsymbol{U}_h$ whenever the corresponding linear coefficient in the modal expansion in Equation (45) exceeds a given threshold value. Here we apply slope limiting when $|\widetilde{C}_i - C_i| > 10^{-6}\,C_i$, for any $\boldsymbol{i}$ satisfying $|\boldsymbol{i}| = d+1$. (C_i and $\widetilde{C}_i$ are arbitrary components of the vectors $\boldsymbol{C}_i$ and $\widetilde{\boldsymbol{C}}_i$, respectively.) In Equation (55), the parameter $\beta_{\rm Tvd}$ takes values in the closed interval [1, 2] and determines how aggressively to apply limiting. The minimal $\beta_{\rm Tvd}$ corresponds to a TVD scheme, which is more dissipative than a scheme with the maximal $\beta_{\rm Tvd}$, which is potentially more oscillatory. Increasing $\beta_{\rm Tvd}$ puts more weight on the neighboring cell averages, making the minmod function more likely to set $\widetilde{\boldsymbol{C}}_i = \boldsymbol{C}_i$, which results in no limiting being applied. The superscripts $-/+$ on the $\boldsymbol{C}_1$ coefficients in the minmod function in Equation (55) indicate that the coefficient belongs to the expansion in the previous/next element in the coordinate direction of the slope to be limited. Figure 2 illustrates how the minmod limiter works in the 1D case when applied to a scalar field $U(x)$. The transformation matrix $\mathcal{M}$ is included in Equation (55) to allow for limiting in characteristic fields (see discussion below). For componentwise limiting, $\mathcal{M}$ is set to the identity matrix. Thus, when slope limiting is applied, the local solution is truncated as

$$\boldsymbol{U}_h(\boldsymbol{x},\,t) := \widetilde{\boldsymbol{U}}_h(\boldsymbol{x},\,t) = \sum_{\substack{\boldsymbol{i}=\boldsymbol{1} \\ |\boldsymbol{i}|\leqslant d+1}}^{\boldsymbol{N}} \widetilde{\boldsymbol{C}}_{\boldsymbol{i}}(t)\,\tilde{\phi}_{\boldsymbol{i}}(\boldsymbol{x}), \tag{57}$$

where $\widetilde{\boldsymbol{C}}_{\boldsymbol{i}} = 0$ for all $\boldsymbol{i}$ with $|\boldsymbol{i}| > d+1$, and

$$\widetilde{\boldsymbol{C}}_{\boldsymbol{1}} := \boldsymbol{U}_{\boldsymbol{K}} - \frac{1}{V_{\boldsymbol{K}}} \sum_{\substack{\boldsymbol{i}=\boldsymbol{1} \\ |\boldsymbol{i}|=d+1}}^{\boldsymbol{N}} \int_{\boldsymbol{K}} \tilde{\phi}_{\boldsymbol{i}}(\boldsymbol{x})\,dV_h\,\widetilde{\boldsymbol{C}}_{\boldsymbol{i}}. \tag{58}$$

Thus, the minmod limiter reduces the local polynomial degree to at most $k = 1$. If the arguments in the minmod function in Equation (55) have different signs, the minmod limiter further reduces the polynomial degree to $k = 0$. Because of this, we use the TCI as discussed below. Although not considered for `thornado` yet, we note that it is possible to generalize or improve the limiting strategy to maintain a higher order of accuracy; see, e.g., Biswas et al. (1994), Krivodonova (2007), Dumbser et al. (2014).

The readjustment of $\widetilde{\boldsymbol{C}}_{\boldsymbol{1}}$ in Equation (58), which occurs after computing the limited slopes in Equation (55), is necessary to preserve the cell average as defined in Equation (31) and is due to the use of curvilinear coordinates (see also related discussion by Radice & Rezzolla 2011, their Section C1). Preservation of the cell average in the limiting procedure is needed, e.g., to conserve mass. Without the "conservative correction" in Equation (58), the limiter preserves the cell average defined in Equation (47), which is undesirable in curvilinear coordinates. Note that the second term on the right-hand side of Equation (58) vanishes in Cartesian coordinates because of the orthogonality of the Legendre polynomials. However, in curvilinear coordinates, this term does not vanish because the Legendre polynomials are not orthogonal with respect to the inner product weighted by $\sqrt{\gamma_h}$. In practice, we have found that the conservative correction is small but necessary to maintain conservation to machine precision.

We note that, in order to improve the evolution of the electron fraction, $Y_{\rm e} = D_{\rm e}/\rho$, we also apply the minmod limiter directly to the electron fraction, and enforce limiting of both ρ_h and $D_{{\rm e},h}$ whenever oscillations in $Y_{\rm e}$ are detected by the minmod function.

In order to determine where slope limiting is necessary, we use the TCI of Fu & Shu (2017) to prevent excessive limiting. For example, it is well known that the minmod limiter is overly diffusive around smooth extrema, where $\widetilde{\boldsymbol{C}}_{\boldsymbol{i}} = 0$, which kills off all the high-order accuracy. We note in passing that other TCIs have been proposed (see, e.g., Qiu & Shu 2005), but we have chosen the one by Fu & Shu (2017) for its relative ease of implementation. This TCI is based on the function

$$I_{\boldsymbol{K}}(G_h) = \frac{\sum_j |G_{\boldsymbol{K}} - G_{\boldsymbol{K}}^{(j)}|}{\max(\max_j |G_{\boldsymbol{K}^{(j)}}^{(j)}|,\,|G_{\boldsymbol{K}}|)}, \tag{59}$$

where $G_h \in \boldsymbol{G}_h \subseteq \boldsymbol{U}_h$ is in the subset of fields used to detect troubled cells. In Equation (59), the sum in the numerator is taken over all the neighboring elements $\boldsymbol{K}^{(j)}$ sharing a boundary with the target element $\boldsymbol{K}$, while the max in the denominator is taken over neighboring elements $\boldsymbol{K}^{(j)}$ and the target element $\boldsymbol{K}$. The cell average of G_h in $\boldsymbol{K}$ is denoted $G_{\boldsymbol{K}}$, and is here given by the right-hand side of Equation (47)—i.e., without the weighting factor $\sqrt{\gamma_h}$ used in the proper definition of the cell average in Equation (31). Computed in the same way, $G_{\boldsymbol{K}}^{(j)}$ is the corresponding cell average computed by extrapolating the polynomial representation from the neighboring elements $\boldsymbol{K}^{(j)}$ into the target $\boldsymbol{K}$, and $G_{\boldsymbol{K}^{(j)}}^{(j)}$ is the cell average native to the neighbor element $\boldsymbol{K}^{(j)}$. An illustration of the TCI is given in Figure 3 for the 1D case applied to a single field $G(x)$.

An element is flagged for limiting if, for any $G_h \in \boldsymbol{G}_h$, $I_{\boldsymbol{K}}(G_h) > C_{\rm TCI}(G)$, where $C_{\rm TCI}(G)$ is a user-defined threshold, which can be set differently for each G. In the numerical results

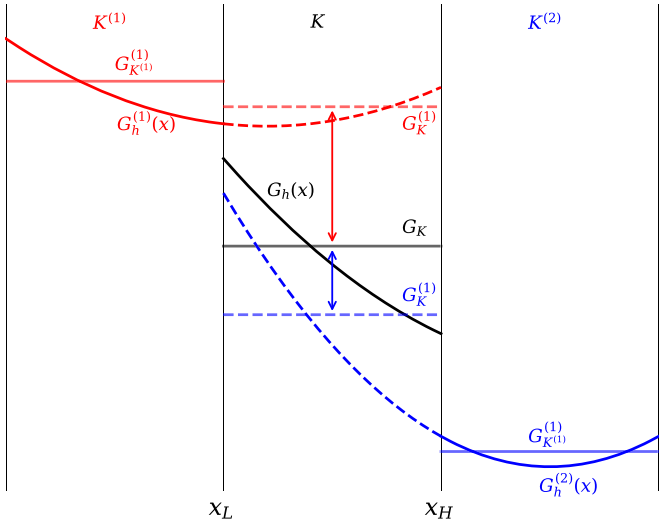

Figure 3. Illustration of how the troubled-cell indicator works in the 1D case on a scalar field $G(x)$ to determine if limiting is needed in the target element K, where the polynomial representation is given by $G_h(x)$ (solid black curve), and the cell average is G_K (solid gray line). The polynomial representation in the left element, $K^{(1)}$, is given by $G_h^{(1)}(x)$ (solid red curve), with cell average $G^{(1)}_{K^{(1)}}$ (solid light red line). Similarly, the polynomial representation in the right element, $K^{(2)}$, is given by $G_h^{(2)}(x)$ (solid blue curve), with cell average $G^{(2)}_{K^{(2)}}$ (solid light blue line). The extrapolations of $G_h^{(1)}(x)$ and $G_h^{(2)}(x)$ into the target element are given by the dashed red and blue curves, respectively. Finally, the cell averages of the extrapolations of $G_h^{(1)}(x)$ and $G_h^{(2)}(x)$, computed over the target cell, are denoted $G_K^{(1)}$ (dashed light red line) and $G_K^{(2)}$ (dashed light blue line), respectively. The element is flagged for limiting if the difference in the cell averages, $|G_K - G_K^{(2)}|$ and/or $|G_K - G_K^{(2)}|$, becomes too large.

presented in Section 4, we use the mass density, fluid energy, and electron fraction as the variables to detect troubled cells; i.e., $\boldsymbol{G} = (\rho, E, Y_e)^{\mathrm{T}}$.

When solving a system of hyperbolic conservation laws, experience has shown that the slope limiting described above is more efficient when performed on the so-called "characteristic variables," as opposed to the conserved variables $\boldsymbol{U}_h$ (see, e.g., Cockburn & Shu 1998, for a description). Because the Euler equations form a system of hyperbolic partial differential equations (see, e.g., LeVeque 1992), the flux Jacobian in Equation (10) can be decomposed as

$$\frac{\partial \boldsymbol{F}^i(\boldsymbol{U})}{\partial \boldsymbol{U}} = \mathcal{R}^i \, \Lambda^i \, (\mathcal{R}^i)^{-1} \quad (i = 1,\ldots, d), \tag{60}$$

where the columns of the 6×6 matrix $\mathcal{R}^i$ contain the right eigenvectors of the flux Jacobian, the rows of $(\mathcal{R}^i)^{-1}$ contain the left eigenvectors, and Λ^i is a diagonal matrix containing the eigenvalues of the flux Jacobian. For hyperbolic systems, the eigenvalues are real and the eigenvectors form a complete set (see e.g., LeVeque 1992). At this point, we introduce the characteristic variable $\boldsymbol{w} = \mathcal{R}^{-1}\boldsymbol{U}$. Recall in Equation (55) that we introduced the transformation matrix $\mathcal{M}$. If we let $\mathcal{M} = (\mathcal{R}^i)^{-1}$, limiting is performed on the characteristic variables. (For linear systems, the characteristic variables evolve independently, and limiting of one characteristic variable does not affect the others.) Once $\mathcal{M}\,\widetilde{\boldsymbol{C}}_i$ is estimated in the characteristic variables as in Equation (55), the limited slopes in the conserved variables are obtained by left multiplication with $\mathcal{M}^{-1}$ (see e.g., Cockburn & Shu 1998; Schaal et al. 2015), and the limiting process proceeds as in Equations (57) and (58). It should be noted that $\mathcal{R}^i$ and $(\mathcal{R}^i)^{-1}$ are computed using cell averages of the conserved and metric variables.

While this process of characteristic limiting has been done for an ideal EoS and shown (e.g., Schaal et al. 2015) to give superior results when compared to componentwise limiting (especially for the high-order case; $k \geqslant 1$), the extension to the tabulated nuclear EoS case is nontrivial. The reasons for this are (1) the increased complexity and dimensionality of the system due to the added electron conservation equation in Equation (4), and (2) the additional care that must be taken when computing the thermodynamic derivatives associated with the flux Jacobian. In the case of an ideal, or other simplified EoS, the necessary thermodynamic derivatives (such as derivatives of pressure) are analytically defined. For a nuclear EoS, the derivatives do not have analytic expressions and the necessary eigenvectors must be constructed generally. We provide the characteristic decomposition of the flux Jacobian for the Euler system with a nuclear EoS in Appendix A.

3.4. Bound-enforcing Limiting

When solving the Euler equations of gas dynamics with an ideal EoS, the mass density ρ and pressure p (or, equivalently, internal energy density e) must remain positive. However, this property is not guaranteed by the basic DG method, which encourages the use of a more advanced procedure (Zhang & Shu 2010a). The internal energy density is given in terms of the conserved quantities as

$$e(\boldsymbol{U}) = E - \frac{m^2}{2\rho}, \tag{61}$$

where $m^2 = m_j m^j$, E is the fluid energy density, and $m_j = \rho v_j$ are the components of the momentum density. For the ideal EoS case, the set of physically admissible states is given by

$$\tilde{\mathcal{G}} = \{\boldsymbol{U} = (\rho, \boldsymbol{m}, E)^{\mathrm{T}} \mid \rho > 0 \text{ and } e(\boldsymbol{U}) > 0 \}. \tag{62}$$

If the mass density is positive, the internal energy density is a concave function of $\boldsymbol{U}$, and $\tilde{\mathcal{G}}$ is a convex set (Zhang & Shu 2010a). For many EoSs (including the ideal EoS), where the pressure only depends on the mass density and internal energy density, $\boldsymbol{U}$ must remain in $\tilde{\mathcal{G}}$ as defined in Equation (62), otherwise, the initial value problem is ill posed. To maintain $\boldsymbol{U} \in \tilde{\mathcal{G}}$, the combination of a suitable time-step restriction, a strong stability-preserving time integrator, and a bound-enforcing limiter is used (e.g., Zhang & Shu 2010a). The time-step restriction is derived as a sufficient condition to ensure that the updated cell average satisfies $\boldsymbol{U}_{\boldsymbol{K}}^{n+1} \in \tilde{\mathcal{G}}$ and requires $\boldsymbol{U}_h^n \in \tilde{\mathcal{G}}$ pointwise within each element, while the limiter, which relies on $\boldsymbol{U}_{\boldsymbol{K}}^{n+1} \in \tilde{\mathcal{G}}$ *and* the convexity of $\tilde{\mathcal{G}}$, is used to again enforce pointwise $\boldsymbol{U}_h^{n+1} \in \tilde{\mathcal{G}}$ within each element. (We do not attempt to derive a sufficient time-step restriction for the present setting in this paper and simply use the condition in Equation (44).) We note here that for two arbitrary elements $\boldsymbol{U}_a, \boldsymbol{U}_b \in \tilde{\mathcal{G}}$, because the set $\tilde{\mathcal{G}}$ is convex, the convex combination $\boldsymbol{U}_c := \vartheta\, \boldsymbol{U}_a + (1-\vartheta)\boldsymbol{U}_b$, where $\vartheta \in [0, 1]$, is also in $\tilde{\mathcal{G}}$; i.e., $\boldsymbol{U}_c \in \tilde{\mathcal{G}}$. Moreover, $e(\boldsymbol{U})$ in Equation (61) is concave because Jensen's inequality—$e(\boldsymbol{U}_c) \geqslant \vartheta\, e(\boldsymbol{U}_a) + (1-\vartheta) e(\boldsymbol{U}_b)$—holds. The property of convex combinations is commonly used to

design constraint-preserving numerical methods for systems where—for physical reasons—the dynamics is constrained to a convex set (see, e.g., Xing et al. 2010; Olbrant et al. 2012; Endeve et al. 2015; Wu & Tang 2015; Chu et al. 2019, for examples beyond the nonrelativistic Euler equations with an ideal EoS).

To maintain physically admissible states in the present setting with `thornado`, we draw inspiration from the limiting strategy proposed for an ideal EoS by Zhang & Shu (2010a), which we have modified to work satisfactorily with a tabulated nuclear EoS. Specifically, thermodynamic quantities, including the specific internal energy $\epsilon = e/\rho$, are tabulated in terms of mass density, temperature, and electron fraction, which cover finite extents, i.e., $\rho \in [\rho_{\rm min}, \rho_{\rm max}]$, $T \in [T_{\rm min}, T_{\rm max}]$, and $Y_{\rm e} \in [Y_{e,\rm min}, Y_{e,\rm max}]$. We use some of the table bounds to define the set of admissible states as

$$\mathcal{G} = \{\boldsymbol{U} = (\rho, \boldsymbol{m}, E, D_{\rm e})^{\rm T} \mid (\rho, D_{\rm e})^{\rm T} \in \mathcal{G}_{\boldsymbol{u}} \text{ and } \epsilon(\boldsymbol{U}) \geqslant \epsilon_{\rm min} \equiv \epsilon(\rho, T_{\rm min}, Y_{\rm e})\}, \tag{63}$$

where we have defined the subset

$$\mathcal{G}_{\boldsymbol{u}} = \{\boldsymbol{u} = (\rho, D_{\rm e})^{\rm T} \mid \rho_{\rm min} \leqslant \rho \leqslant \rho_{\rm max}, D_{\rm e} > 0, \text{ and } Y_{\rm e,min} \leqslant Y_{\rm e}(\boldsymbol{u}) \leqslant Y_{\rm e,max}\}, \tag{64}$$

and seek to maintain $\boldsymbol{U}_h \in \mathcal{G}$.

First, we note that it is straightforward to show that the subset $\mathcal{G}_{\boldsymbol{u}}$ is convex. To do this, it is sufficient to show that a convex combination of two arbitrary elements of $\mathcal{G}_{\boldsymbol{u}}$ also belongs to $\mathcal{G}_{\boldsymbol{u}}$. To this end, let $\boldsymbol{u}_a \equiv (\rho_a, D_{{\rm e},a})^{\rm T} \in \mathcal{G}_{\boldsymbol{u}}$, $\boldsymbol{u}_b \equiv (\rho_b, D_{{\rm e},b}) \in \mathcal{G}_{\boldsymbol{u}}$, and define the convex combination $\boldsymbol{u}_c = \vartheta\, \boldsymbol{u}_a + (1-\vartheta)\boldsymbol{u}_b$, where $\vartheta \in [0, 1]$. Then the first component of $\boldsymbol{u}_c$ is $\rho_c = \vartheta\, \rho_a + (1-\vartheta)\rho_b$. Because, by assumption, $\rho_a, \rho_b \in [\rho_{\rm min}, \rho_{\rm max}]$ and $\vartheta \in [0, 1]$, it follows that $\rho_c \in [\rho_{\rm min}, \rho_{\rm max}]$. Similarly, the second component of $\boldsymbol{u}_c$ is $D_{{\rm e},c} = \vartheta\, D_{{\rm e},a} + (1-\vartheta)D_{{\rm e},b}$. Then, because $D_{{\rm e},a}$, $D_{{\rm e},b} > 0$, it follows that $D_{{\rm e},c} > 0$. Finally, we can write

$$\begin{aligned} Y_{\rm e}(\boldsymbol{u}_c) &= \frac{D_{{\rm e},c}}{\rho_c} = \frac{\vartheta\, D_{{\rm e},a} + (1-\vartheta)D_{{\rm e},b}}{\vartheta\, \rho_a + (1-\vartheta)\rho_b} \\ &= \alpha\, \frac{D_{{\rm e},a}}{\rho_a} + (1-\alpha)\frac{D_{{\rm e},b}}{\rho_b} \\ &= \alpha\, Y_{\rm e}(\boldsymbol{u}_a) + (1-\alpha) Y_{\rm e}(\boldsymbol{u}_b), \end{aligned} \tag{65}$$

where

$$\alpha = \frac{\vartheta\, \rho_a}{\vartheta\, \rho_a + (1-\vartheta)\rho_b}. \tag{66}$$

Because $\rho_a, \rho_b \geqslant \rho_{\rm min} > 0$ and $\vartheta \in [0, 1]$, we have $\alpha \geqslant 0$. We also have $\alpha \leqslant 1$. Therefore, $\alpha \in [0, 1]$, which implies $Y_{\rm e}(\boldsymbol{u}_c) \in [Y_{\rm e,min}, Y_{\rm e,max}]$ and $\boldsymbol{u}_c \in \mathcal{G}_{\boldsymbol{u}}$. Thus, the subset $\mathcal{G}_{\boldsymbol{u}}$ is convex.

While, strictly speaking, the Euler equations in Section 2 are valid for any mass density $\rho > 0$, we note that there are physical reasons for maintaining the mass density within the finite table bounds, which are $\rho_{\rm min} \approx 1.66 \times 10^3$ g cm^{-3} and $\rho_{\rm max} \approx 3.16 \times 10^{15}$ g cm^{-3} for the tables used in this paper. Indeed, in CCSN simulations, it is possible for the cell-averaged mass density to evolve outside these limits, which would require extending the table bounds. However, when the mass density approaches the upper bound, a relativistic description should be adopted, and when the mass density approaches the lower bound, the nuclear EoS adopted here is invalid because the matter is not in NSE. These bounds must, however, also be enforced to avoid algorithm failure. For the purpose of the bound-enforcing limiter, the finite bounds on the mass density in Equation (63) are included in case the bounds are violated for certain points within an element, e.g., in the vicinity of a shock, while the cell-averaged mass density is still inside the table bounds. (The limiter developed here will not work if the cell-averaged mass density exceeds the table bounds.) We have also equipped the set of admissible states $\mathcal{G}$ with the bounds $Y_{\rm e} \in [Y_{\rm e,min}, Y_{\rm e,max}] \subseteq [0, 1]$, which are also required to avoid algorithm failure. (In this work, $Y_{\rm e,min} = 0.01$ and $Y_{\rm e,max} = 0.7$). We note, however, that for the test problems in Section 4 and the application in Section 5, we did not encounter a situation in which the mass density or the electron fraction exceeded their respective table bounds.

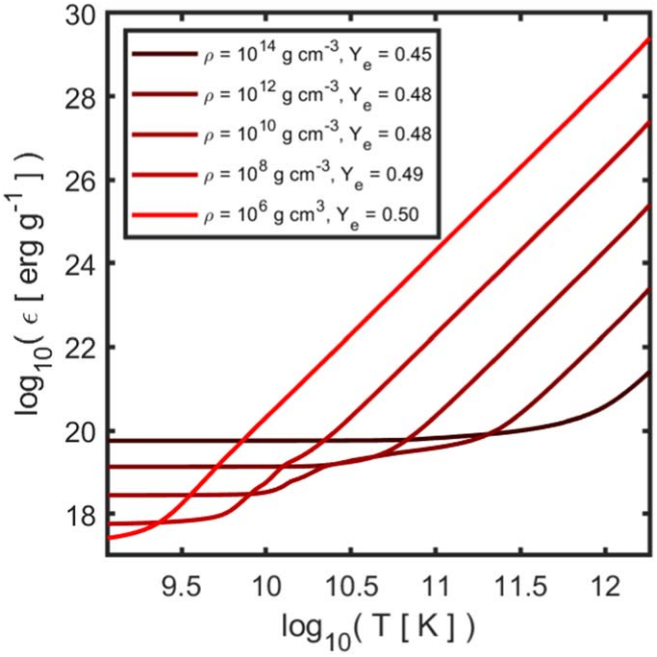

Figure 4. Relationship between specific internal energy and temperature from the SFHo EoS for select values of mass density and electron fraction. (Note that the electron fraction is only sampled from the narrow range encountered in the adiabatic simulations discussed in Section 5.) Due to degeneracy, the specific internal energy for all profiles demonstrates asymptotic behavior for low temperatures. Thus, the lower boundary on ϵ would not change if the table was reconstructed with a lower temperature limit.

On the other hand, a complication that frequently arises in gravitational collapse simulations is that the specific internal energy falls below the minimum tabulated value (i.e., $\epsilon < \epsilon_{\rm min}$)—especially around core bounce and shock formation, which we discuss in further detail in Section 5. When this happens, the EoS is not invertible for the temperature when given the state vector $(\rho, \epsilon, Y_{\rm e})^{\rm T}$, and the algorithm fails because the temperature is needed to compute the pressure as well as other thermodynamic quantities. It is not feasible to merely generate tables with lower $T_{\rm min}$, because—particularly for high mass densities—the specific internal energy does not tend to zero as $T \to 0$ due to the degeneracy (or zero temperature) contribution to the internal energy, as can be seen in Figure 4. In CCSN simulations, where the iron core is degenerate at the onset of collapse, the initial specific internal energy is already close to the minimum value. Then, around core bounce and shock

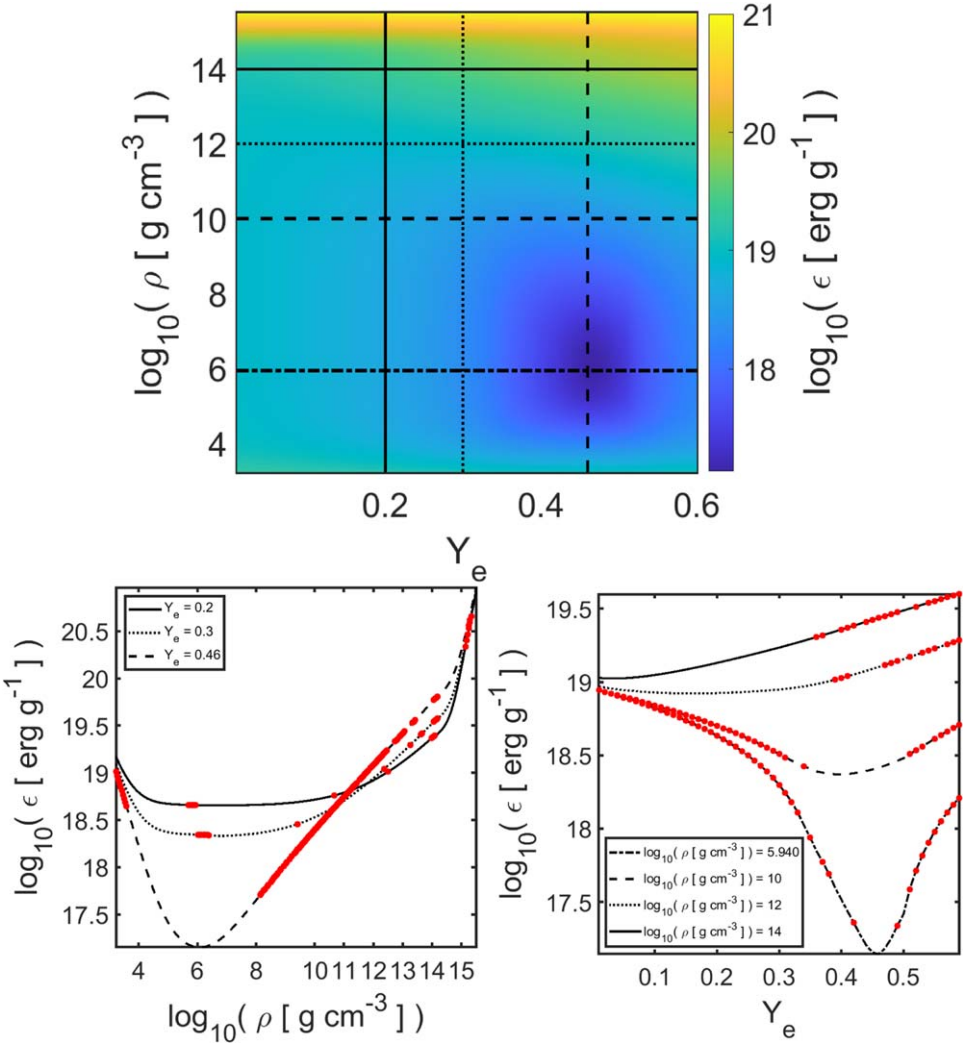

Figure 5. The minimum specific internal energy, $\epsilon_{\rm min} = \epsilon(\rho, T_{\rm min}, Y_{\rm e})$ for the SFHo EoS, is displayed in the top panel as a function of ρ and $Y_{\rm e}$. (Because the specific internal energy can be negative, we have added an offset of about 2.8×10^{17} erg g^{-1} to the actual value returned by the EoS.) The vertical lines (constant $Y_{\rm e}$) correspond to lines plotted in the lower-left panel, while horizontal lines (constant ρ) correspond to lines plotted in the lower-right panel. To further highlight the topology of the $\epsilon_{\rm min}$ surface, we plot $\epsilon_{\rm min}$ vs. mass density for select values of $Y_{\rm e}$ in the lower-left panel. Similarly, we plot $\epsilon_{\rm min}$ vs. electron fraction for select values of ρ in the lower-right panel. These traces are selected to illustrate the nonconvexity of $\mathcal{G}$. The red dots indicate nonconvex regions, i.e., where the second derivative of the specific internal energy with respect to either ρ (left) or $Y_{\rm e}$ (right) is less than zero. From visual inspection, the constant $Y_{\rm e}$ profiles (lower left) appear to be convex; i.e., $(\partial^2 \epsilon_{\rm min}/\partial \rho^2)_{Y_{\rm e}} \geqslant 0$. However, the dashed line is not, and the dotted and solid lines are nonconvex around $\rho = 10^{14}$ g cm^{-3}. Meanwhile, the constant ρ profiles (lower right) are clearly not all convex because $(\partial^2 \epsilon_{\rm min}/Y_{\rm e}^2)_{\rho}$ can be negative (see dashed and dashed–dotted curves).

formation, where steep gradients in the evolved fields form, conditions with $\boldsymbol{U}_h \notin \mathcal{G}$ can easily arise within certain elements, and a limiting strategy is needed. Fortunately, we have observed that $\boldsymbol{U}_{\boldsymbol{K}} \in \mathcal{G}$ is always satisfied (although we do not seek to establish sufficient conditions to guarantee this here). This allows us to pursue the limiting strategy proposed by Zhang & Shu (2010a), which we detail below.

There is, however, an additional complication that may cause the limiting strategy of Zhang & Shu (2010a) to fail: the surface of specific internal energy at the minimum temperature $T_{\rm min}$—that is, $\epsilon_{\rm min}(\rho, Y_{\rm e}) \equiv \epsilon(\rho, T_{\rm min}, Y_{\rm e})$—is not globally convex in the sense that the second derivatives $(\partial^2 \epsilon_{\rm min}/\partial \rho^2)_{Y_{\rm e}}$ and $(\partial^2 \epsilon_{\rm min}/\partial Y_{\rm e}^2)_{\rho}$ are not strictly positive everywhere, which implies that the set $\mathcal{G}$ in Equation (63) is not strictly convex. We illustrate this in Figure 5, which shows $\epsilon(\rho, T_{\rm min}, Y_{\rm e})$ as a function of ρ and $Y_{\rm e}$ for the SFHo EoS (Steiner et al. 2013b). Therefore, adopting the limiting procedure from the ideal EoS case to enforce $\boldsymbol{U}_h \in \mathcal{G}$—even if $\boldsymbol{U}_{\boldsymbol{K}} \in \mathcal{G}$—can compromise the robustness of the limiter. The reason is that the amount of limiting applied to the polynomial $\boldsymbol{U}_h$ is determined by finding the intersection point of the boundary of $\mathcal{G}$ and the straight line connecting the cell average $\boldsymbol{U}_{\boldsymbol{K}}$ and a nonphysical point value $\boldsymbol{U}_{\boldsymbol{q}} \notin \mathcal{G}$. If $\mathcal{G}$ is not convex, there may be multiple intersection points, which can cause the limiter to fail. However, the issue of a globally nonconvex $\mathcal{G}$ is avoided if the limiter is only activated in regions for which $\mathcal{G}$ is locally convex. That is, for the elements that require limiting, the cell average $\boldsymbol{U}_{\boldsymbol{K}}$ and the DG solution $\boldsymbol{U}_h$, evaluated in the required quadrature points within each element $\boldsymbol{K}$, are in a locally convex region and sufficiently close to each other in $\mathcal{G}$. The latter is typically the case in regions of the flow characterized by small gradients but may not be the case in the vicinity of a shock. Fortunately, as discussed further in Section 5, we do not encounter any situations in which the nonconvexity of $\mathcal{G}$ causes the limiter to fail, but this needs to be further investigated in the context of multidimensional models with higher physical fidelity (i.e.,

models that include neutrino transport), which sample a larger part of the EoS than the simulations discussed in this paper.

The bound-enforcing limiter is completely local to each element and can thus be discussed in terms of a single element $\boldsymbol{K}$. As in (Zhang & Shu 2010a), we define a point set $\boldsymbol{S}^+$, which includes the volumetric nodal points in an element $\boldsymbol{K}$, as well as the points on the interface of $\boldsymbol{K}$. For the 2D case with $k=2$, the point set is given by the union of all the points displayed in the right panel in Figure 1. Thus, $\boldsymbol{S}^+$ comprises the points where $\boldsymbol{U}_h$ is evaluated to construct the update for each $\boldsymbol{U}_p$ in Equation (30). Using Equation (17), the solution is evaluated at all the points $\boldsymbol{x}_q \in \boldsymbol{S}^+$, and limiting is applied if, for any point $\boldsymbol{x}_q \in \boldsymbol{S}^+$, $\boldsymbol{U}_q \equiv \boldsymbol{U}_h(\boldsymbol{x}_q) \notin \mathcal{G}$. The step-by-step procedure for bound-enforcing limiting is described next, where it is assumed that the cell average satisfies $\boldsymbol{U}_K \in \mathcal{G}$.

3.4.1. Step 1: Mass Density and Electron Density

The first step is to enforce $\rho_q \in [\rho_{\min}, \rho_{\max}]$ and $D_{\mathrm{e},q} \geqslant \delta_{D_\mathrm{e}} > 0$ for all $\boldsymbol{x}_q \in \boldsymbol{S}^+$, where δ_{D_e} is arbitrarily small. (The bound $D_{\mathrm{e},q} > 0$ is needed in Step 2 below.) Following Zhang & Shu (2010b), we use the linear scaling limiter from Liu & Osher (1996), and replace the polynomial $\boldsymbol{u}_h(\boldsymbol{x}) = (\rho_h(\boldsymbol{x}), D_{\mathrm{e},h}(\boldsymbol{x}))^{\mathrm{T}}$ with the limited polynomial

$$\boldsymbol{u}_h^{(1)}(\boldsymbol{x}) \coloneqq (1-\vartheta_1)\boldsymbol{u}_K + \vartheta_1\,\boldsymbol{u}_h(\boldsymbol{x}) \quad (\vartheta_1 \in [0,1]), \tag{67}$$

where the limiter parameter $\vartheta_1 \in [0, 1]$ is found by a simple backtracing algorithm. Specifically, for any point $\boldsymbol{x}_q \in \boldsymbol{S}^+$ with $\boldsymbol{u}_q = \boldsymbol{u}_h(\boldsymbol{x}_q) \notin \mathcal{G}_u$, we start with $\vartheta_{1,q} = 1$, which is recursively reduced (by 5%) until

$$\boldsymbol{u}_q^{(1)} = (1-\vartheta_{1,q})\boldsymbol{u}_K + \vartheta_{1,q}\,\boldsymbol{u}_q \in \mathcal{G}_u. \tag{68}$$

(In practice, to reduce the number of iterations, we set $\vartheta_{1,q} = 0$ whenever the backtracing algorithm has brought the value below 0.01.) We then set $\vartheta_1 \coloneqq \min_q \vartheta_{1,q}$, where the minimum is taken over all the points within the element where $\boldsymbol{u}_h$ was found to violate the bounds associated with Step 1. The limiter in Equation (67) simply scales $\boldsymbol{u}_h$ as evaluated in the points within the element toward the cell average, and the value for ϑ_1 is determined in order to scale the solution in the points just enough to ensure that the bounds are satisfied for all $\boldsymbol{x}_q \in \boldsymbol{S}^+$. In the worst-case scenario, $\vartheta_1 = 0$, and the DG solution is set equal to the cell average everywhere within the element. Note that this step is conservative and does not change the cell averages; i.e., $\boldsymbol{u}_K^{(1)} = \boldsymbol{u}_K$. Also note that if the bounds on the mass density and electron density are not violated, then $\vartheta_1 = 1$ and $\boldsymbol{u}_h^{(1)}(\boldsymbol{x}) = \boldsymbol{u}_h(\boldsymbol{x})$.

3.4.2. Step 2: Electron Fraction

In the second step, we enforce $Y_{\mathrm{e},q} \equiv D_{\mathrm{e},h}(\boldsymbol{x}_q)/\rho_h(\boldsymbol{x}_q) \in [Y_{\mathrm{e,min}}, Y_{\mathrm{e,max}}]$ for all $\boldsymbol{x}_q \in \boldsymbol{S}^+$. To do this, we follow a procedure similar to the previous step, and replace $\boldsymbol{u}_h^{(1)}(\boldsymbol{x})$ with the limited polynomial

$$\boldsymbol{u}_h^{(2)}(\boldsymbol{x}) \coloneqq (1-\vartheta_2)\boldsymbol{u}_K + \vartheta_2\,\boldsymbol{u}_h^{(1)}(\boldsymbol{x}) \quad (\vartheta_2 \in [0,1]), \tag{69}$$

where

$$\vartheta_2 = \frac{\alpha\,\rho_K}{\alpha\,\rho_K + (1-\alpha)\rho_\alpha^{(1)}} \quad \text{and}$$
$$\alpha = \min\left\{1, \left|\frac{Y_{\mathrm{e,min}} - Y_{\mathrm{e},K}}{m_{Y_\mathrm{e}} - Y_{\mathrm{e},K}}\right|, \left|\frac{Y_{\mathrm{e,max}} - Y_{\mathrm{e},K}}{M_{Y_\mathrm{e}} - Y_{\mathrm{e},K}}\right|\right\}, \tag{70}$$

and where we have defined

$$M_{Y_\mathrm{e}} = \max_{\boldsymbol{x}\in\boldsymbol{S}^+} Y_\mathrm{e}(\boldsymbol{u}_h^{(1)}(\boldsymbol{x})), \quad m_{Y_\mathrm{e}} = \min_{\boldsymbol{x}\in\boldsymbol{S}^+} Y_\mathrm{e}(\boldsymbol{u}_h^{(1)}(\boldsymbol{x})), \quad \text{and}$$
$$Y_{\mathrm{e},K} = D_{\mathrm{e},K}/\rho_K. \tag{71}$$

and with the cell average for mass density and electron density computed according to the definition in Equation (31), i.e.,

$$\rho_K = \frac{\sum_{p=1}^N w_p\,\sqrt{\gamma_p}\,\rho_p}{\sum_{p=1}^N w_p\,\sqrt{\gamma_p}} \quad \text{and} \quad D_{\mathrm{e},K} = \frac{\sum_{p=1}^N w_p\,\sqrt{\gamma_p}\,D_{\mathrm{e},p}}{\sum_{p=1}^N w_p\,\sqrt{\gamma_p}}, \tag{72}$$

respectively. In the expression for ϑ_2 in Equation (70), we simply set $\rho_\alpha^{(1)} = \max_{x\in\boldsymbol{S}^+}\rho_h^{(1)}(\boldsymbol{x})$, which is sufficient, but may not give the optimal value for ϑ_2 (i.e., this choice may not give the largest ϑ_2 while still maintaining $\boldsymbol{u}_h^{(2)} \in \mathcal{G}_u$).

Step 2 is also conservative and does not change the cell averages; i.e., $\boldsymbol{u}_K^{(2)} = \boldsymbol{u}_K^{(1)} = \boldsymbol{u}_K$. Also, if the bounds on the electron fraction are not violated, $\vartheta_2 = \alpha = 1$ and $\boldsymbol{u}_h^{(2)}(\boldsymbol{x}) = \boldsymbol{u}_h^{(1)}(\boldsymbol{x})$. After the completion of Steps 1 and 2, we have ensured $\boldsymbol{u}_h^{(2)}(\boldsymbol{x}_q) \in \mathcal{G}_u$ for all $\boldsymbol{x}_q \in \boldsymbol{S}^+$.

3.4.3. Step 3: Specific Internal Energy

In the third, and final, step, we enforce $\epsilon_q \geqslant \epsilon_{\min,q}$ for all $\boldsymbol{x}_q \in \boldsymbol{S}^+$. To this end, we define $\boldsymbol{U}_h^{(2)} = (\rho_h^{(2)}, \boldsymbol{m}_h, E_h, D_{\mathrm{e},h}^{(2)})^{\mathrm{T}}$, which is the full solution vector after steps 1 and 2. Using $\boldsymbol{U}_h^{(2)}$, the specific internal energy and electron fraction in each point $\boldsymbol{x}_q \in \boldsymbol{S}^+$ are computed as

$$\epsilon_q^{(2)} = \epsilon(\boldsymbol{U}_h^{(2)}(\boldsymbol{x}_q)) = \left(E_q - \frac{m_q^2}{2\,\rho_q^{(2)}}\right)/\rho_q^{(2)} \quad \text{and}$$
$$Y_{\mathrm{e},q}^{(2)} = Y_\mathrm{e}(\boldsymbol{U}_h^{(2)}(\boldsymbol{x}_q)) = D_{\mathrm{e},q}^{(2)}/\rho_q^{(2)}, \tag{73}$$

respectively. Then, if $\epsilon_q^{(2)} < \epsilon_{\min,q}^{(2)} \equiv \epsilon(\rho_q^{(2)}, T_{\min}, Y_{\mathrm{e},q}^{(2)})$ for any $\boldsymbol{x}_q \in \boldsymbol{S}^+$, we replace $\boldsymbol{U}_h^{(2)}$ with the limited polynomial

$$\boldsymbol{U}_h^{(3)}(\boldsymbol{x}) \coloneqq (1-\vartheta_3)\boldsymbol{U}_K + \vartheta_3\,\boldsymbol{U}_h^{(2)}(\boldsymbol{x}) \quad (\vartheta_3 \in [0,1]). \tag{74}$$

Here, the polynomial representation of the full solution is written as a convex combination of the cell average and the polynomial representation after Step 2. Because we assume $\boldsymbol{U}_K \in \mathcal{G}$, setting $\vartheta_3 = 0$ will ensure $\boldsymbol{U}_h^{(3)}(\boldsymbol{x}) \in \mathcal{G}$. However, setting $\vartheta_3 = 0$, so that $\boldsymbol{U}_h^{(3)} = \boldsymbol{U}_K$, kills off all the high-order accuracy of the polynomial representation, which is undesirable. Instead, one would want to find the largest value for ϑ_3 to retain as much high-order accuracy as possible and enforce $\boldsymbol{U}_h^{(3)}(\boldsymbol{x}) \in \mathcal{G}$ for all $\boldsymbol{x}_q \in \boldsymbol{S}^+$. As discussed above, this is complicated by the fact that $\mathcal{G}$ is not strictly convex. It is further complicated by the fact that the surface $\epsilon_{\min}(\rho, Y_\mathrm{e})$ is only available at discrete points from the EoS table. Because of this,

we will assume that $\mathcal{G}$ is locally convex and first obtain $\vartheta_{3,\boldsymbol{q}}$ by solving

$$\epsilon\left(\boldsymbol{s}(\vartheta_{3,\boldsymbol{q}})\right) = (1 - \vartheta_{3,\boldsymbol{q}})\,\epsilon_{\min,\boldsymbol{K}} + \vartheta_{3,\boldsymbol{q}}\,\epsilon^{(2)}_{\min,\boldsymbol{q}}, \tag{75}$$

for each $\boldsymbol{x}_{\boldsymbol{q}}$ where $\epsilon^{(2)}_{\boldsymbol{q}} < \epsilon^{(2)}_{\min,\boldsymbol{q}}$. On the left-hand side of Equation (75), we have defined

$$\boldsymbol{s}(\vartheta_{3,\boldsymbol{q}}) = (1 - \vartheta_{3,\boldsymbol{q}})\boldsymbol{U}_{\boldsymbol{K}} + \vartheta_{3,\boldsymbol{q}}\,\boldsymbol{U}^{(2)}_{\boldsymbol{q}}, \tag{76}$$

while on the right-hand side of Equation (75), we have defined $\epsilon_{\min,\boldsymbol{K}} = \epsilon\left(\rho_{\boldsymbol{K}}, T_{\min}, Y_{e,\boldsymbol{K}}\right)$. Then, we set

$$\vartheta_3 \coloneqq \min_{\boldsymbol{q}} \vartheta_{3,\boldsymbol{q}}, \tag{77}$$

where the minimum is taken over all the points in $\boldsymbol{S}^{+}$ where the specific internal energy fell below the minimum value.

We note that the limiter in Equation (74) is conservative in all the fields in the sense that the cell average is preserved, i.e.,

$$\begin{aligned}\frac{1}{V_{\boldsymbol{K}}}\int_{\boldsymbol{K}} \boldsymbol{U}^{(3)}_h\, dV_h &= (1 - \vartheta_3)\boldsymbol{U}_{\boldsymbol{K}} + \vartheta_3\,\frac{1}{V_{\boldsymbol{K}}}\int_{\boldsymbol{K}} \boldsymbol{U}^{(2)}_h\, dV_h \\ &= (1 - \vartheta_3)\boldsymbol{U}_{\boldsymbol{K}} + \vartheta_3\,\boldsymbol{U}_{\boldsymbol{K}} = \boldsymbol{U}_{\boldsymbol{K}}.\end{aligned} \tag{78}$$

The motivation for solving Equation (75) is as follows (see Zhang & Shu 2010a, for the ideal EoS case): $\boldsymbol{s}(\vartheta_{3,\boldsymbol{q}})$ is the parameterized straight line connecting the cell average $\boldsymbol{U}_{\boldsymbol{K}}$ and the point value $\boldsymbol{U}^{(2)}_{\boldsymbol{q}}$. Because $\boldsymbol{U}_{\boldsymbol{K}} \in \mathcal{G}$, we know that

$$\epsilon_{\boldsymbol{K}} \equiv \left(E_{\boldsymbol{K}} - \frac{m^2_{\boldsymbol{K}}}{2\rho_{\boldsymbol{K}}}\right)/\rho_{\boldsymbol{K}} \geqslant \epsilon_{\min,\boldsymbol{K}}. \tag{79}$$

On the other hand, if $\boldsymbol{U}^{(2)}_{\boldsymbol{q}} \notin \mathcal{G}$, there is at least one intersection point of the line $\boldsymbol{s}(\vartheta_{3,\boldsymbol{q}})$ and the boundary of $\mathcal{G}$; i.e., the surface $\epsilon_{\min}(\rho, Y_e)$. (If $\mathcal{G}$ is convex, which we assume in this step, there is exactly one intersection point.) Because we do not know the exact shape of the surface, we approximate it by the line segment connecting the boundary points $\epsilon_{\min,\boldsymbol{K}}$ and $\epsilon^{(2)}_{\min,\boldsymbol{q}}$, and by the convexity assumption, this line lies above the surface $\epsilon_{\min}(\rho, Y_e)$. Thus, in Equation (75), the solution $\vartheta_{3,\boldsymbol{q}}$ provides the intersection point between the line connecting the points $\epsilon_{\boldsymbol{K}}$ and $\epsilon^{(2)}_{\boldsymbol{q}}$ and the line connecting the points $\epsilon_{\min,\boldsymbol{K}}$, $\epsilon^{(2)}_{\min,\boldsymbol{q}}$. See Figure 6 for an illustration.

Equation (75) is solved for $\vartheta_{3,\boldsymbol{q}}$ with a simple bisection algorithm, using the end points $\vartheta_{3,\boldsymbol{q}} = 0$ and $\vartheta_{3,\boldsymbol{q}} = 1$ as starting points. We note that, in practice, the solution to Equation (75) does not have to be accurate to many significant digits, and the bisection algorithm can be terminated after a few iterations. We also note that because $\epsilon_{\min}(\rho, Y_e)$ is not strictly convex, as is shown in Figure 5, Equation (75) can have multiple roots, and the bisection algorithm may result in a limited solution that is still outside $\mathcal{G}$. We have, however, not encountered a situation where this happens. On the contrary, in the numerical examples presented in Section 4, we find that the limiting procedure discussed in this section significantly improves the robustness of the DG algorithm. As can be seen by looking ahead to Figure 20 in Section 5, the bound-enforcing limiter is continuously activated, with $\vartheta_3 \in [0.4, 1]$, in a short time interval around core bounce in an adiabatic collapse simulation.

Finally, we have assumed that the cell average satisfies $\boldsymbol{U}_{\boldsymbol{K}} \in \mathcal{G}$ when the limiter is applied. If this assumption does not hold, the bound-enforcing limiter will fail. By considering the equation for the cell average in Equation (32), in combination with forward Euler time stepping, it may be possible to derive a sufficient restriction on the time step such that $\boldsymbol{U}^{n+1}_{\boldsymbol{K}} \in \mathcal{G}$, provided $\boldsymbol{U}^{n}_{\boldsymbol{K}} \in \mathcal{G}$ and $\boldsymbol{U}^{n}_{\boldsymbol{q}} \in \mathcal{G}$ (possibly with additional points included in the set $\boldsymbol{S}^{+}$). We do, however, not pursue this endeavor here. Instead, we use the time-step restriction given in Equation (44), which may not be sufficient. In the absence of an explicit expression for a sufficient time-step restriction (assuming one exists), one may design a time-step control algorithm where the step size is recursively reduced, and the time step retaken, until a physically admissible cell average is obtained. On the other hand, we have yet to encounter an application in which a solution with cell average $\boldsymbol{U}_{\boldsymbol{K}} \notin \mathcal{G}$ is passed to the bound-enforcing limiter.

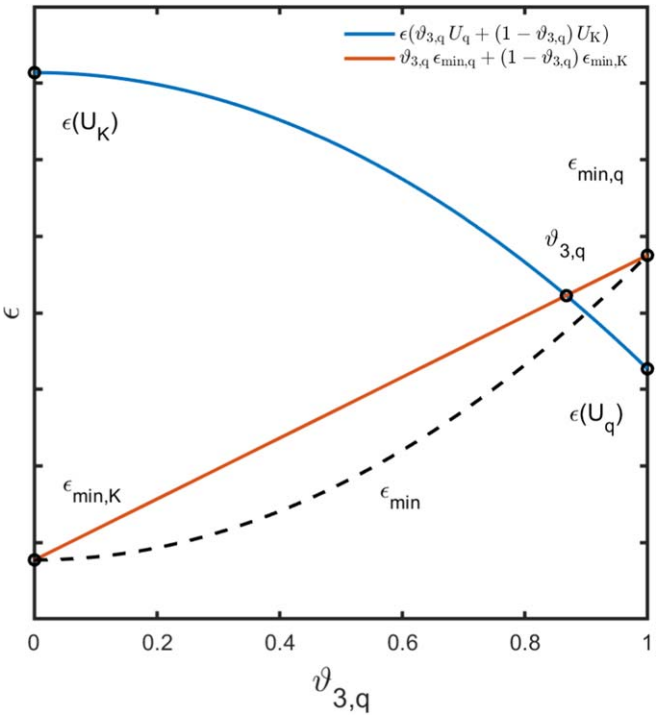

Figure 6. Illustration of the bisection problem used to find $\vartheta_{3,\boldsymbol{q}}$ in Equation (75) to determine the extent of limiting needed to ensure that the specific internal energy does not fall below the table boundary $\epsilon_{\min}$ (dashed black curve). In the example depicted here, $\epsilon(\boldsymbol{U}_{\boldsymbol{q}})$, the right end point of the blue curve, is below the table boundary, and limiting is needed. We find $\vartheta_{3,\boldsymbol{q}}$ as the intersection point between the blue curve connecting $\epsilon(\boldsymbol{U}_{\boldsymbol{K}})$ and $\epsilon(\boldsymbol{U}_{\boldsymbol{q}})$, and the red curve connecting $\epsilon_{\min,\boldsymbol{K}}$ and $\epsilon_{\min,\boldsymbol{q}}$. In this case, $\vartheta_{3,\boldsymbol{q}} \approx 0.87$.

3.5. Poisson Solver

In `thornado`, the approximate Newtonian gravitational potential, Φ_h, is obtained using the Poseidon code (N. Roberts et al. 2021, in preparation). Poseidon solves Equation (5) on a spherical-polar grid with a combination of an angular spectral expansion using spherical harmonics and a radial finite-element solution method. Here, we discuss the case of spherical symmetry and thus limit the angular expansion to the monopole harmonic function. Therefore, we will focus only on the finite-element method (Larson & Bengzon 2013) used in the radial expansion. Because the Newtonian gravitational potential is expected to be continuous in space, we require the approximate solution, Φ_h, to be C^0 continuous across element interfaces. To enforce this continuity, Poseidon uses the continuous Galerkin (CG) finite-element method instead of the DG method to solve the Poisson equation. However, we note that the DG method

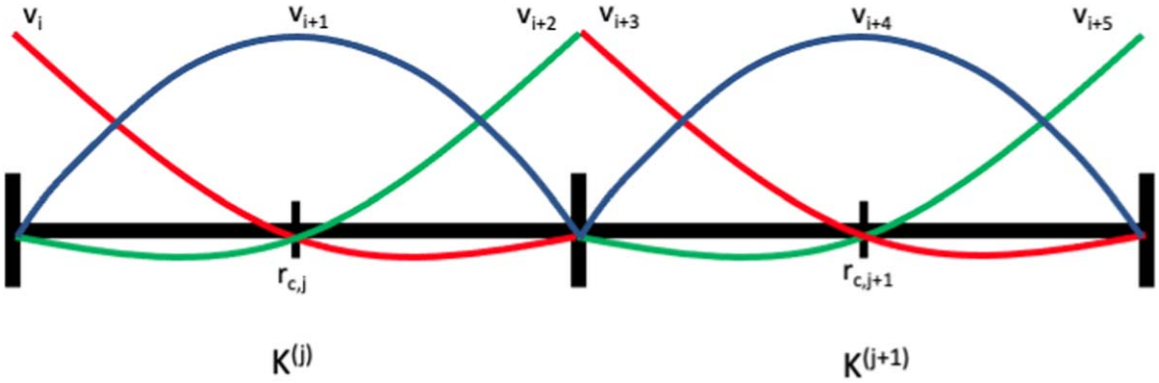

Figure 7. Illustration of the basis functions, v_i, used in the CG solution method of Poseidon for the case $k = 2$. Each function is associated with a specific element $K^{(j)}$ and a node $\xi_{j,m}$ within that element, such that $v_i(r(\xi_{j,m})) = 1$. Outside of the associated element, $v_i = 0$, and is therefore not depicted.

can also be used to solve elliptic equations (e.g., Rivière 2008; Vincent et al. 2019).

The CG method expresses the approximate solution, Φ_h, to Equation (5) as a continuous expansion of functions of the form

$$\Phi_h(r, t) = \sum_{i=1}^{N_D} \Phi_i(t) v_i(r), \tag{80}$$

where N_D is the total number of interpolation nodes on the domain D, and $\Phi_i(t)$ are spatially constant expansion coefficients. As the method used to solve the Poisson equation is purely spatial in nature, we will omit the time parameter, t, for the rest of this section. The basis functions $v_i(r)$ belong to the approximation space, V_h, defined by

$$V_h = \{\psi_h\colon \psi_h|_{K^{(j)}} \in P^k(K^{(j)}), j = 1,\ldots,N_e\}, \tag{81}$$

where P^k is a space of 1D piecewise polynomials of degree k, and $K^{(j)}$ are the radial elements of the same decomposition of the computational domain as expressed in Section 3.1. Given this choice of approximation space and domain decomposition, N_D is given by $N_D = N_e k + 1$, where N_e is the number of radial elements on the domain.

Continuity is achieved through the choice of interpolation points and approximation space polynomials. Within a specific element $K^{(j)}$, the interpolation points, $\hat{S}_j = \{\xi_{j,1},\ldots, \xi_{j,m}, \ldots, \xi_{j,k+1}\} \subset I = \left[-\frac{1}{2}, \frac{1}{2}\right]$, are chosen to be the LGL points. The physical coordinate r is related to the reference coordinate $\xi \in I$ by the transformation

$$r(\xi) = r_{c,j} + \Delta r_j \xi, \tag{82}$$

where $r_{c,j}$ is the physical coordinate for the center of element $K^{(j)}$ and is such that

$$r(\xi_{j,k+1}) = r(\xi_{j+1,1}). \tag{83}$$

The inverse relationship,

$$\xi(r) = \frac{(r - r_{c,j})}{\Delta r_j}, \tag{84}$$

allows us to express the chosen approximation space polynomials $v_i(r) \in V_h$ as

$$v_i(r) = \begin{cases} \ell_{j,m}(\xi(r)) & \text{for } r \in K^{(j)} \\ 0 & \text{else,} \end{cases} \tag{85}$$

where $\ell_{j,m}$ are the Lagrange polynomials in Equation (15) constructed with the LGL points, $\hat{S}_j$. Each approximation function $v_i(r)$ is associated with a node $\xi_{j,m}$ such that $v_i(r(\xi_{j,m})) = 1$ by the Kronecker delta property of the Lagrange polynomials. This choice of interpolation points and approximation functions enforces the C^0 continuity of the solution. See Figure 7 for an illustration of elements and associated basis functions in the CG method for the case with $k = 2$.

The CG method seeks to find $\Phi_h \in V_h$, which approximates Φ in Equation (5) such that

$$\langle \nabla^2 \Phi_h, \psi_h \rangle_D = \langle 4\pi G\rho, \psi_h \rangle_D \tag{86}$$

holds for all test functions $\psi_h \in V_h$. In Equation (86),

$$\langle \nabla^2 \Phi_h, \psi_h \rangle_D = 4\pi \int_{R_L}^{R_H} \nabla^2 \Phi_h \, \psi_h \, r^2 dr, \tag{87}$$

and

$$\langle 4\pi G\rho, \psi_h \rangle_D = 16\pi^2 G \int_{R_L}^{R_H} \rho \, \psi_h \, r^2 dr, \tag{88}$$

where R_L and R_H are the low and high radial boundary locations of the domain, respectively. Using integration by parts on Equation (87), Equation (86) becomes the weak form of Equation (5),

$$-\langle \partial_r \Phi_h, \partial_r \psi_h \rangle_D + (\partial_r \Phi_h)\psi_h|_{R_L}^{R_H} = \langle 4\pi G\rho, \psi_h \rangle_D. \tag{89}$$

For the gravitational collapse problem discussed in Section 5, we impose the Neumann boundary condition,

$$\partial_r \Phi_h(R_L) = 0, \tag{90}$$

on the inner boundary ($R_L = 0$) to preserve the symmetry of the solution, and the Dirichlet boundary condition,

$$\Phi_h(R_H) = -\frac{GM_{enc}}{R_H}, \tag{91}$$

on the outer boundary, where M_{enc} is the total enclosed mass given by

$$M_{enc} = 4\pi \int_{R_L}^{R_H} \rho_h(r) r^2 dr. \tag{92}$$

The Neumann condition in Equation (90) reduces Equation (89) to

$$-\langle \partial_r \Phi_h, \partial_r \psi_h \rangle_D + (\partial_r \Phi_h(R_H))\psi_h(R_H) = \langle 4\pi G\rho, \psi_h \rangle_D. \tag{93}$$

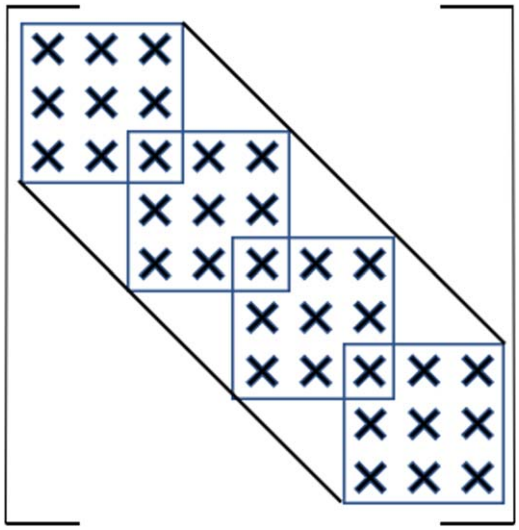

Figure 8. Nonzero structure of the matrix S used in Poseidon for the case of $k = 2$, and $N_e = 4$. The diagonal lines denote the band structure of the matrix. The squares denote the overlapping block structure within the band. The single overlapping element shared by the consecutive blocks comes from the shared interpolation node at element interfaces.

Next, the expansion in Equation (80) and $\psi_h = v_j$ are substituted into Equation (93) to give

$$\sum_{i=1}^{N_D} \Phi_i(-\langle \partial_r v_i, \partial_r v_j\rangle_D + (\partial_r v_i(R_{\rm H}))v_j(R_{\rm H})) = \langle 4\pi G\rho, v_j\rangle_D, \quad j = 1,\cdots,N_D. \tag{94}$$

To enforce the Dirichlet condition, the expansion coefficient Φ_N is set to the boundary value given by Equation (91), and the dimensionality of the problem is reduced to $N_D - 1$, eliminating the $(\partial_r v_i(R_{\rm H}))v_j(R_{\rm H})$ term as $v_j(R_{\rm H}) = 0, \ \forall \ j \neq N_D$. Equation (94) then becomes

$$\sum_{i=1}^{N_D-1} -\Phi_i\langle \partial_r v_i, \partial_r v_j\rangle_D = \langle 4\pi G\rho, v_j\rangle_D, \quad j = 1,\cdots,N_D - 1. \tag{95}$$

Defining the stiffness matrix as

$$S = \{s_{ij}\}_{i,j=1}^{N_D-1}, \ \ s_{ij} = -\langle \partial_r v_i, \partial_r v_j\rangle_D, \tag{96}$$

the load vector as

$$L = \{\langle 4\pi G\rho, v_j\rangle_D\}_{j=1}^{N_D-1}, \tag{97}$$

and the unknown coefficient vector as

$$C = \{\Phi_i\}_{i=1}^{N_D-1}, \tag{98}$$

the system in Equation (95) can then be written in matrix form as

$$SC = L. \tag{99}$$

The matrix S is a sparse symmetric band matrix, with bandwidth equal to k. When $k = 1$, the matrix S is tridiagonal. When $k > 1$, an overlapping block structure occurs within the diagonal band of S; see Figure 8.

The sparsity of the matrix is given by

$$\text{Sparsity} = \frac{N_e k^2 - N_e + 1}{N_e^2 k^2 + 2N_e k + 1}. \tag{100}$$

To reduce memory overhead, S is stored in compressed column storage (CCS) format. The system is then solved using a CCS-compatible Cholesky factorization. Once these coefficients are known, the approximate solution can be reconstructed anywhere within the domain using Equation (80).

3.6. Table Interpolation

As in Bruenn (1985), Mezzacappa & Messer (1999), and Bruenn et al. (2020), we obtain a thermodynamic quantity F and its derivatives from the tabulated EoS through trilinear interpolation in the space spanned by $(\log_{10}(\rho), \log_{10}(T), Y_{\rm e})$. The software to compute these are provided by the WeakLib library, and, for completeness, we restate formulas here. To simplify the notation, let $X = \log_{10}(\rho)$, $Y = \log_{10}(T)$, and $Z = Y_{\rm e}$. Then, $\bar{F} = \bar{F}(X, Y, Z)$, where $\bar{F}$ is related to the thermodynamic quantity by $F = 10^{\bar{F}} - F_{\rm offset}$. That is, trilinear interpolations are performed on logged quantities, and the offset is used to ensure that $\bar{F}$ is well defined when F is negative. Obtaining F first requires the eight points (X_a, Y_b, Z_c): $a, b, c \in \{0, 1\}$ from the table that make up the corners of a "cube" of the points closest to (X, Y, Z). These points then satisfy

$$X_1 - X_0 = \frac{1}{N_\rho}, \quad Y_1 - Y_0 = \frac{1}{N_T}, \quad \text{and} \quad Z_1 - Z_0 = \frac{1}{N_{Y_{\rm e}}}, \tag{101}$$

where N_ρ and N_T are the number of the points per decade in ρ and T, respectively, and $N_{Y_{\rm e}}$ is the number of points per unit interval in $Y_{\rm e}$. $\bar{F}$ is then given by the trilinear interpolation formula, e.g., found in Equation (32) in Mezzacappa & Messer (1999), which, in multi-index notation, can be written compactly as

$$\bar{F}(\boldsymbol{X}) = \sum_{\boldsymbol{i}=\boldsymbol{0}}^{\boldsymbol{1}} w_{\boldsymbol{i}}(\boldsymbol{X})\bar{F}_{\boldsymbol{i}}, \tag{102}$$

where $\boldsymbol{X} = (X, Y, Z)$. In this context, the weights $w_{\boldsymbol{i}}(\boldsymbol{X})$ are given by

$$w_{\boldsymbol{i}}(\boldsymbol{X}) = B_{i_1}(X)B_{i_2}(Y)B_{i_3}(Z), \tag{103}$$

where $B_{i_1}(X)$ ($i_1 \in \{0, 1\}$) are linear Lagrange polynomials,

$$B_0(X) = \frac{X_1 - X}{X_1 - X_0} \quad \text{and} \quad B_1(X) = \frac{X - X_0}{X_1 - X_0}, \tag{104}$$

and $B_{i_2}(Y)$ and $B_{i_3}(Z)$ are similarly defined by replacing X with Y or Z, respectively.

As in Mezzacappa & Messer (1999), derivatives with respect to ρ, T, and $Y_{\rm e}$ are calculated directly from this expression, i.e.,

$$\left(\frac{\partial F}{\partial \rho}\right)_{T,Y_{\rm e}} = \frac{(F + F_{\rm offset})}{\rho}\left(\frac{\partial \bar{F}}{\partial X}\right)_{Y,Z} = \frac{(F + F_{\rm offset})}{\rho}\sum_{\boldsymbol{i}=\boldsymbol{0}}^{\boldsymbol{1}} \frac{\partial w_{\boldsymbol{i}}}{\partial X}\bar{F}_{\boldsymbol{i}}, \tag{105}$$

$$\left(\frac{\partial F}{\partial T}\right)_{\rho,Y_e} = \frac{(F + F_{\rm offset})}{T}\left(\frac{\partial \bar{F}}{\partial Y}\right)_{X,Z} = \frac{(F + F_{\rm offset})}{T}\sum_{i=0}^{1}\frac{\partial w_i}{\partial Y}\bar{F}_i, \quad (106)$$

$$\left(\frac{\partial F}{\partial Y_e}\right)_{\rho,T} = (F + F_{\rm offset})\left(\frac{\partial \bar{F}}{\partial Z}\right)_{X,Y} = (F + F_{\rm offset})\sum_{i=0}^{1}\frac{\partial w_i}{\partial Z}\bar{F}_i. \quad (107)$$

We note that this interpolation scheme does not, by construction, satisfy the Maxwell relations of thermodynamics. While this may impact the ability to resolve adiabatic flows (see Swesty 1996 and Timmes & Swesty 2000 for further discussion), we do not observe any clear evidence of this being a problem in our computations. In addition, while we believe that the low-order accuracy of the trilinear interpolation scheme may play a role in both the convergence rates observed with the high-order RKDG scheme in Section 4.1 and the issues with characteristic limiting around the phase transition observed in Section 5, additional investigations are required.

4. Numerical Results

In this section, we present results obtained with the DG method as implemented in thornado for various test problems relevant to CCSNe and other astrophysical phenomena. With the exception of a few reference calculations obtained using an ideal EoS in Section 4.1.1, all of the results were obtained using a tabulated version of the SFHo EoS of Steiner et al. (2013b), which covers the ranges $\rho \in [1.66 \times 10^3, 3.16 \times 10^{15}]$ g cm^{-3}, with $N_\rho = 25$, $T \in [1.16 \times 10^9,\ 1.84 \times 10^{12}]$ K, with $N_T = 50$, and $Y_e \in [0.01, 0.7]$, with $N_{Y_e} = 100$. (See, however, Endeve et al. 2019 for a documentation of results obtained with `thornado` using an ideal EoS.) In the first two subsections, we begin by presenting results from 1D advection tests using Cartesian coordinates and 1D and 2D Riemann problems using Cartesian, spherical-polar, and cylindrical coordinates (Sections 4.1 and 4.2, respectively). These tests serve as an initial gauge of the implementation of the DG algorithm in `thornado` with a nuclear EoS. Using Riemann problems with initial conditions adapted from their ideal EoS counterparts, we aim to investigate the performance of our implementation in curvilinear coordinates, as well as the slope limiter presented in Section 3.3 and the bound-enforcing limiter presented in Section 3.4. The Poisson solver is tested in Section 4.3. Then, in Section 5, our focus turns to the main application, adiabatic gravitational collapse in spherical symmetry, where we investigate the performance of `thornado`'s DG algorithm by investigating various aspects of the solver with an eye toward future spherically symmetric—and eventually multi-dimensional—supernova simulations with neutrino transport. In all of the tests, the CFL number in Equation (44) is set to $C_{\rm CFL} = 0.5$.

4.1. Advection Tests

4.1.1. Rate of Convergence

The accuracy of the DG method can be manipulated by changing the number of nodes per cell $N = k + 1$ and/or the total number of cells K. The number of nodes per cell (or element) governs the expected order of accuracy of the method. (Nth order spatial accuracy is expected with N nodes.) This section covers the rate at which changing the number of degrees of freedom $n_{\rm DOF} = (k+1) \times K$ impacts the accuracy, i.e., the convergence rate. Inspired by Suresh & Huynh (1997), this test is performed over the 1D computational domain $D = [-100,\ 100]$ km, with smooth initial conditions and periodic boundary conditions. The initial state for the tabulated EoS case is set with the primitive state vector $\boldsymbol{P}$ as

$$\boldsymbol{P} = (\rho,\, u,\, p,\, Y_e)^{\rm T} = (\rho_0(1 + 0.1\,\sin^4(\pi x/L)),\, v_0,\, p_0,\, 0.3)^{\rm T},$$

where $\rho_0 = 10^{12}$ g cm^{-3} is the background density, $v_0 = 0.1c$ is the velocity, $p_0 = 0.01\,\rho_0\,c^2$ the background pressure, and $L = 200$ km is the domain length. In this test, the mass density, a quartic sine wave, is advected for one period without any limiting, while the velocity, pressure, and electron fraction remain constant. The error in mass density between the initial and final states is then calculated in the L_1 error norm,

$$L_1 \equiv \sum_{j=1}^{n_{\rm DOF}} |\rho_{j,\rm final} - \rho_{j,\rm initial}|. \quad (108)$$

In Figure 9, we plot this quantity, scaled by both $n_{\rm DOF}$ and a background density ρ_0, versus $n_{\rm DOF}$ (crosses). (For reference, we also plot results obtained with an ideal EoS case with $\Gamma = 1.4$; open circles.) The solutions are obtained using $N = 2$ (black symbols) and $N = 3$ (red symbols) nodes with second- and third-order time-integration schemes, respectively. For each N, we use seven different values of K: 8, 16, 32, 64, 128, 256, and 512. For this smooth problem, we always observe that for a fixed $n_{\rm DOF}$, the scheme with $N = 3$ is significantly more accurate than the scheme with $N = 2$. For the nuclear EoS case, the L_1 error for the second-order scheme ($N = 2$) crosses zero

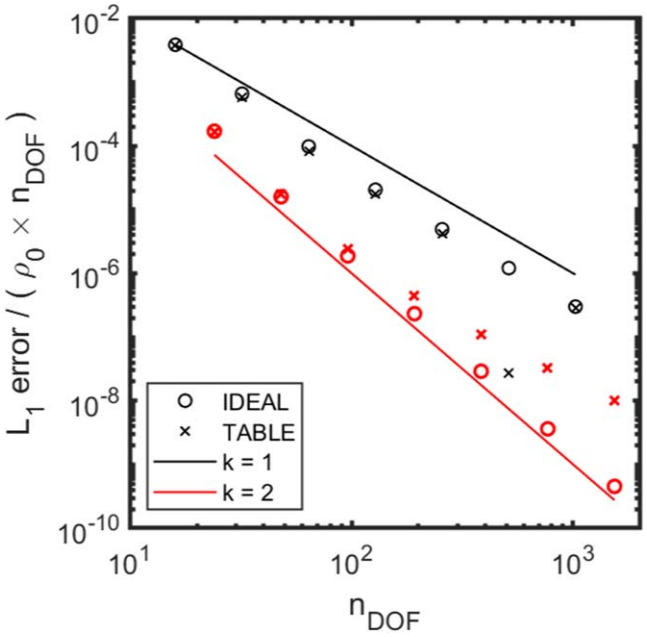

Figure 9. L_1 error between the initial and final states of an advected quartic sine wave, adopted from Suresh & Huynh (1997). The results are scaled by the number of degrees of freedom to obtain the average error per node. For the tabulated EoS results, the background density $\rho_0 = 10^{12}$ g cm^{-3} is also used to scale the error, but $\rho_0 = 1$ for the ideal case, which is run in dimensionless mode. The solid lines are proportional to $n_{\rm DOF}^{k+1}$ and serve as references for the convergence rates of solutions represented by polynomials of degree $k = 1$ and $k = 2$.

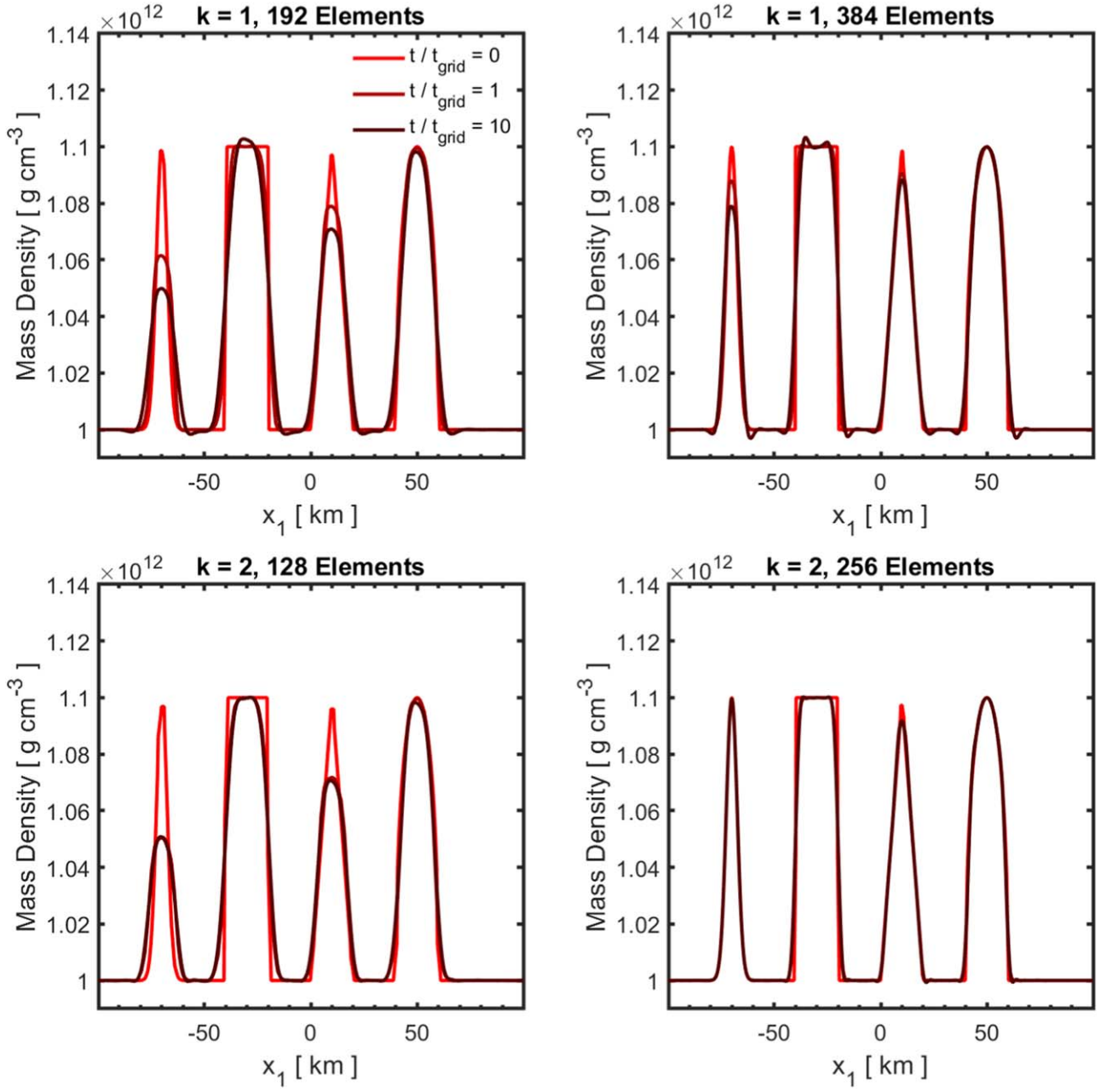

Figure 10. Mass density profiles for the discontinuous multiwave advection test adopted from Suresh & Huynh (1997). In each panel, we plot the initial condition ($t/t_{\rm grid} = 0$; light red), the solution after 1 period ($t/t_{\rm grid} = 1$; medium red) and after 10 periods ($t/t_{\rm grid} = 10$; dark red). ($t_{\rm grid}$ is the physical time required for one grid crossing.) In the top panels, we plot results obtained with the second-order method ($k = 1$ and second-order SSP-RK time stepping) using 192 (left panel) and 384 (right panel) elements. In the bottom panels, we plot results obtained with the third-order method ($k = 2$ and third-order SSP-RK time stepping) using 128 (left panel) and 256 (right panel) elements. Increasing the number of nodes and/or elements results in better resolution around sharp peaks.

and generates a cusp at $n_{\rm DOF} = 512$. Otherwise, the results obtained with the second-order method agree well with the expected convergence rate for both the tabulated and ideal EoS cases. For $N = 3$, the ideal EoS case exhibits third-order accuracy throughout. However, for $n_{\rm DOF} > 96$, the results for $N = 3$ with the tabulated EoS appear to undergo a transition from third-order to second-order accuracy. We suspect that the trilinear interpolation method discussed in Section 3.6 may be the cause of the loss of accuracy for large $n_{\rm DOF}$, but this requires further investigation.

4.1.2. Discontinuous Multiwave

This test from Suresh & Huynh (1997) involves the advection of a discontinuous initial state for mass density, which includes a Gaussian wave, a square wave, a triangular wave, and a semiellipse (see light red lines in Figure 10). This test is performed over a periodic 1D domain $D = [-100, 100]$ km, with the initial state given as

$$\boldsymbol{P} = (\rho, u, p, Y_{\rm e})^{\rm T} = (\rho(x, 0), v_0, p_0, 0.3)^{\rm T},$$

where v_0 and p_0 are given the same values as in the previous test, and $\rho(x, 0)$ is a piecewise function defined as

$$\begin{aligned}
&\rho(x, 0) = \rho_0(1 + 0.1\exp(-\log(2)(x/L + 0.7)^2/(0.0009)))\\
&\quad \text{if } -80\text{ km} \leqslant x \leqslant -60\text{ km}\\
&\rho(x, 0) = \rho_0(1 + 0.1)\\
&\quad \text{if } -40\text{ km} \leqslant x \leqslant -20\text{ km}\\
&\rho(x, 0) = \rho_0(1 + 0.1(1 - |10(x/L - 0.1)|))\\
&\quad \text{if } \quad 0\text{ km} \leqslant x \leqslant \quad 20\text{ km}\\
&\rho(x, 0) = \rho_0(1 + 0.1(1 - 100(x/L - 0.5)^2)^{1/2})\\
&\quad \text{if } \quad 40\text{ km} \leqslant x \leqslant \quad 60\text{ km}\\
&\rho(x, 0) = \rho_0 \quad \text{otherwise},
\end{aligned}$$

where $L = 100$ km.

We compare the performance of second- and third-order schemes in this test. Thus, a second- and third-order SSP-RK time-integration scheme was used for $k = 1$ and $k = 2$, respectively. This test used the characteristic limiting procedure described in Section 3.3 with a TCI threshold $C_{\rm TCI} = 1.0 \times 10^{-3}$ and a TVD parameter $\beta_{\rm TVD} = 1.5$. Figure 10 shows the initial density profile (light red lines) along with four different cases of the mass density being evolved 1 (medium red lines) and 10 (dark red lines) times across the periodic domain. Results obtained with the second-order method are displayed in the top panels, while results obtained with the third-order method are displayed in the bottom panels. Note that the results displayed in the top-left and top-right panels were obtained using the same total number of degrees of freedom as the results displayed in the bottom-left and bottom-right panels, respectively. Analytically, the evolved solution should match up exactly with the initial condition after each full domain crossing. However, the numerical solution is distorted by dissipation and dispersion. For fixed $n_{\rm DOF}$, the third-order method appears to provide more accurate results. As the solution is evolved in the $k = 1$ case, accuracy is lost primarily around sharp edges, namely, for the Gaussian and triangular waveforms. For the $k = 1$ case with 384 elements, the solution is not well resolved around the base of each waveform, but some accuracy is gained around the maxima. Loss of accuracy around sharp edges is also observed with the third-order method using 128 elements (bottom-left panel). However, as is seen in the bottom-right panel, the features of the solution are better captured with the third-order method using 256 elements. For the third-order method, we note that most of the distortion of the initial profile occurs in the first domain crossing, as the profiles after 1 and 10 crossings are almost on top of each other. This is not so much the case for the second-order scheme, where the results after 1 and 10 crossings are more easily distinguished. However, there is a trade-off between numerical accuracy and computational expense.

4.2. Riemann Problems

4.2.1. Sod Shock Tube: Cartesian Coordinates

This test is based on the classic Riemann problem from Sod (1978). It involves an initially stationary fluid with a discontinuity separating two states—left and right—with high pressure and density on the left and low pressure and density on the right. This initial state evolves into a shock propagating into the low-density region, followed by a contact discontinuity and a rarefaction wave propagating back into the high-density state. Shock-tube problems such as this stress a method's ability to capture shocks and contact discontinuities without smearing or introducing unphysical oscillations. Given the importance of shocks in CCSNe, this serves as a critical first test for any method designed to model these explosions.

Here, the problem is modified to use physical units in a regime realizable in simulations of CCSNe. The computational domain is $D = [-5, 5]$ km with the discontinuity initially at $x = 0$ km, separating the left and right states

$$\boldsymbol{P}_{\rm L} = (\rho, v, p, Y_{\rm e})^{\rm T}_{\rm L} = (10^{12}\text{ g cm}^{-3}, 0, 10^{32}\text{ erg cm}^{-3}, 0.4)^{\rm T} \tag{109}$$

$$\boldsymbol{P}_{\rm R} = (\rho, v, p, Y_{\rm e})^{\rm T}_{\rm R} = (1.25 \times 10^{11}\text{ g cm}^{-3}, 0, 10^{31}\text{ erg cm}^{-3}, 0.3)^{\rm T}. \tag{110}$$

(Note that the initial $Y_{\rm e}$ profile is also discontinuous.)

The problem is evolved until $t = 0.021$ ms, using 100 uniform elements with $\beta_{\rm Tvd} = 1.75$, and no TCI ($C_{\rm TCI} = 0$), so that limiting is applied everywhere. We use third-order spatial discretization ($k = 2$) and third-order temporal integration (SSP-RK3). The main focus of this test is to compare results obtained with componentwise and characteristic limiting (discussed in Section 3.3). Figure 11 shows results for mass density (upper left), pressure (upper right), velocity (lower left), and electron fraction (lower right), using both characteristic (blue) and componentwise limiting (red), compared to a reference solution (black) computed using the first-order accurate spatial method ($k = 0$), third-order time integration, and 10,000 elements. We note that both limiting schemes capture the general nature of the solution, including the rarefaction wave, which extends from about -3 to 0 km, the contact discontinuity, which is located at about 2 km, and the shock, located at about 4 km. The scheme based on characteristic limiting, however, is better at suppressing oscillations, and is less dissipative across the contact discontinuity. These observations are consistent with those made by Schaal et al. (2015) in the ideal EoS case.

4.2.2. Sod Shock Tube: Spherical-polar and Cylindrical Coordinates

As a test of `thornado`'s ability to work with non-Cartesian coordinate systems, we also solve a spherically symmetric version of the Sod shock-tube problem in 1D spherical-polar and 2D cylindrical coordinates. For spherical-polar coordinates, the domain is $D = [0, 10]$ km, with the initial discontinuity placed at $r = 5$ km, while, for cylindrical coordinates, our domain is $D = [0, 10]$ km $\times$ $[-10, 10]$ km, and the discontinuity is placed at $r = \sqrt{R^2 + z^2} = 5$ km. For the initial left and right states, we use those given in the 1D Cartesian Sod test in Equations (109)–(110), with the exception that the electron fraction is given a constant value of $Y_{\rm e} = 0.4$ across the entire domain. We evolve both tests until $t = 0.025$ ms using 100 elements in the spherical case and 100×200 elements in the cylindrical case. Both tests use the third-order methods ($k = 2$ and SSP-RK3), characteristic limiting with $\beta_{\rm Tvd} = 1.75$, and no TCI ($C_{\rm TCI} = 0$). We note that for the 2D test with cylindrical coordinates, we used `thornado`'s interface to AMReX to take advantage of AMReX's MPI infrastructure.

Results are shown in Figure 12. In the left panel of Figure 12, we show the 2D density distribution for the

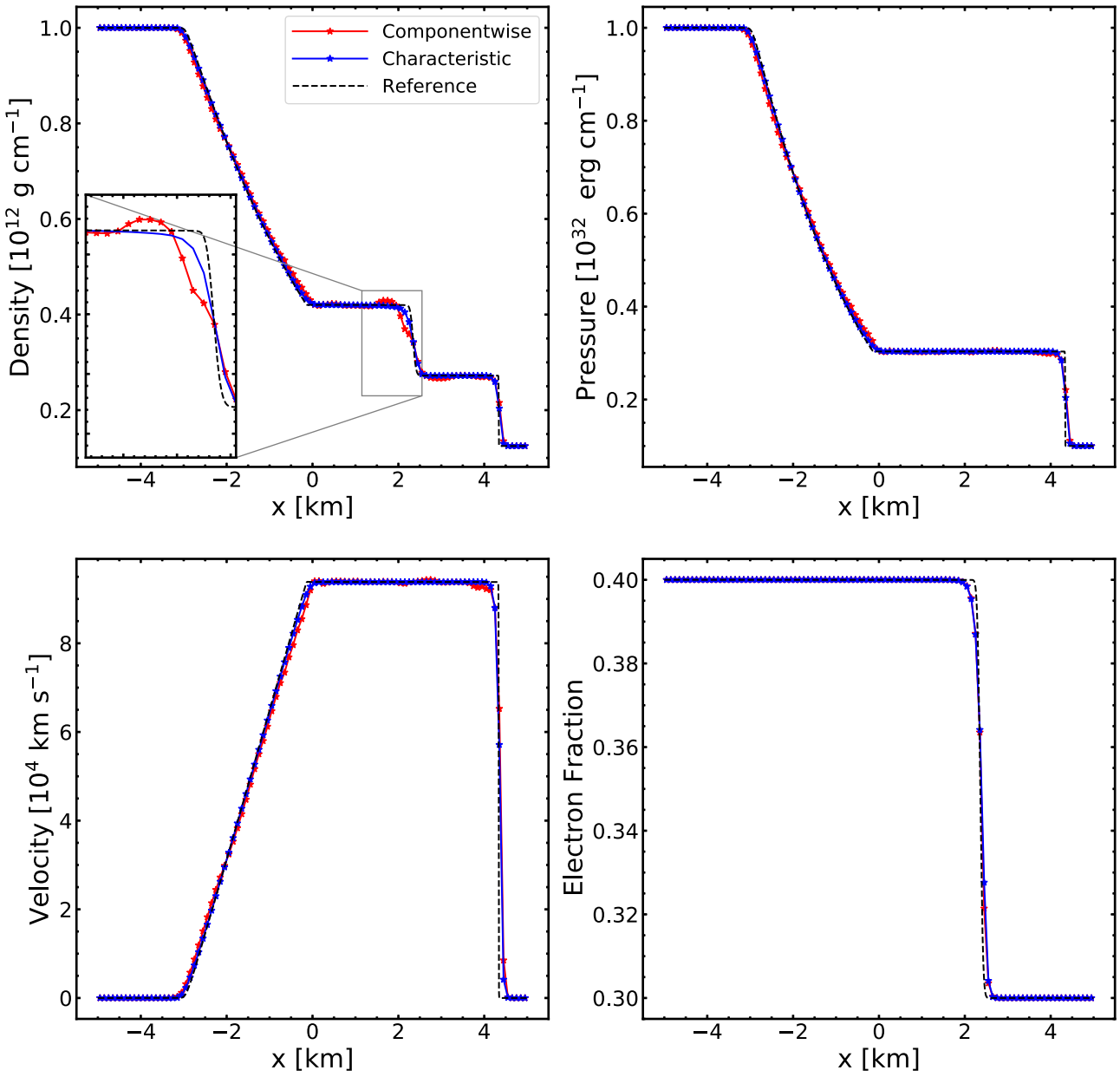

Figure 11. Numerical solution of the Sod shock tube at $t = 0.021$ ms using 100 elements and third-order accurate methods with characteristic (blue) and componentwise limiting (red) for density (upper left), pressure (upper right), velocity (lower left), and electron fraction (lower right), compared to a reference solution (black) using 10,000 elements, obtained with first-order spatial discretization and third-order time integration.

cylindrical test. In the right panel of Figure 12, we show the density, velocity, and pressure profiles of the spherical-polar test (solid lines), along with scatter plots of the corresponding quantities from the cylindrical test versus spherical-polar radius r for comparison. We note that the characteristics of the solution profiles are similar to those obtained by others using an ideal EoS (e.g., Omang et al. 2006). There is also good agreement between the results obtained with spherical-polar and cylindrical coordinates. As in the Cartesian test, we note the clear resolution of the shock and contact discontinuity with no discernible oscillations. Furthermore, we note some spread in the scatter plots from the cylindrical solution, most notably in the velocity profile across the contact discontinuity. However, despite the truly multidimensional setup in the cylindrical case, there is decent preservation of the spherical symmetry inherent in the test.

4.2.3. Shock Tube Provoking the Bound-enforcing Limiter

This test, performed in 1D with Cartesian coordinates, is similar to the Sod shock tube discussed in Section 4.2.1, but with initial conditions tuned to provoke the bound-enforcing limiter developed in Section 3.4. The goal is to demonstrate that the limiter keeps the solution within the set of admissible states (specifically that $\epsilon \geqslant \epsilon_{\rm min}$) while also conserving the total mass, energy, and electron number in time, given, respectively, by

$$\int_D \{\rho_h(x, t), E_h(x, t), D_{{\rm e},h}(x, t)\}\, dx. \tag{111}$$

The computational domain is $D = [-5, 5]$ km, and a discontinuity is placed at $x = 0$ km, which separates the left

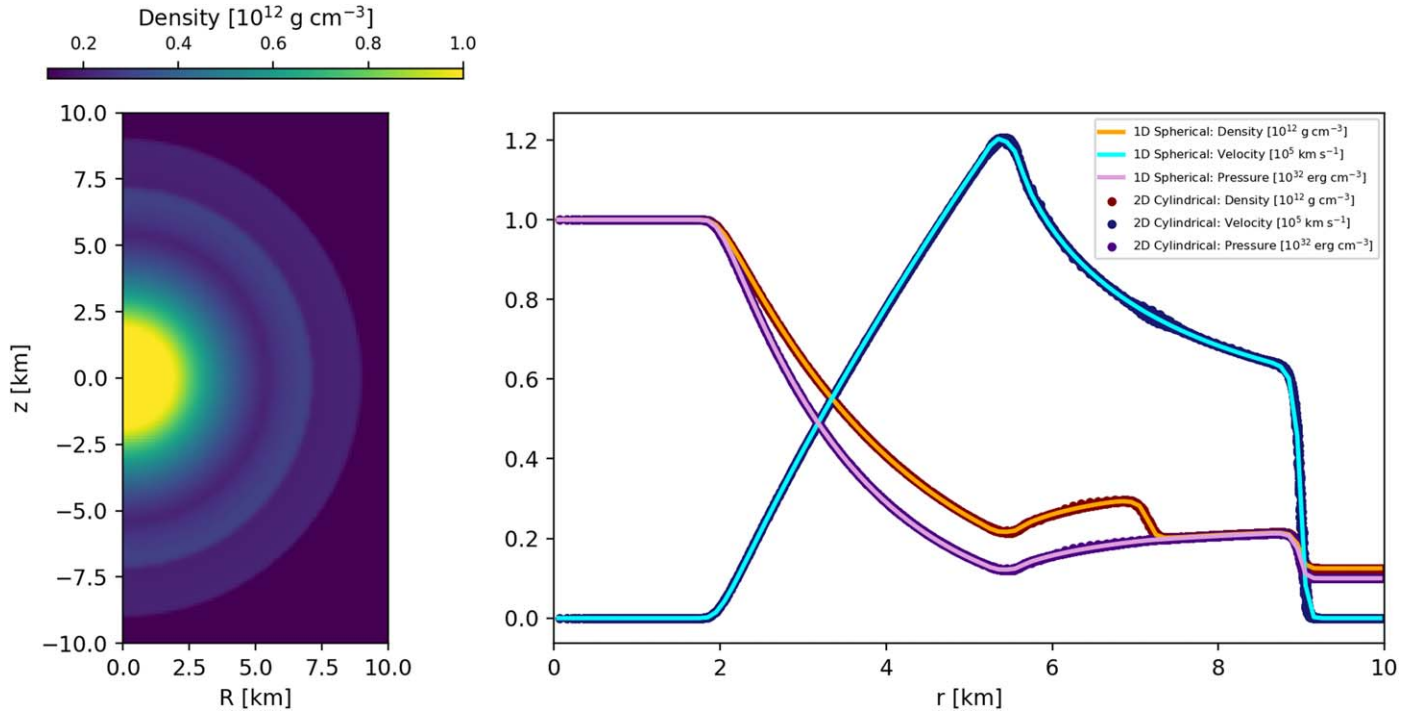

Figure 12. Two-dimensional density distribution (left panel), along with radial density, velocity, and pressure profiles (right panel) for the spherically symmetric Sod shock-tube problem evolved to $t = 0.025$ ms using both 1D spherical-polar and 2D cylindrical coordinates: solid lines and scatter plots, respectively.

and right states of the Riemann problem

$$\boldsymbol{P}_{\rm L} = (\rho,\, v,\, p,\, Y_{\rm e})^{\rm T}_{\rm L} = (1.00 \times 10^{13}\ {\rm g\ cm^{-3}},\ 0\ {\rm m\ s^{-1}},\ 1.070 \times 10^{31}\ {\rm erg\ cm^{-3}},\ 0.04)^{\rm T}$$

$$\boldsymbol{P}_{\rm R} = (\rho,\, v,\, p,\, Y_{\rm e})^{\rm T}_{\rm R} = (1.25 \times 10^{12}\ {\rm g\ cm^{-3}},\ 0\ {\rm m\ s^{-1}},\ 1.023 \times 10^{30}\ {\rm erg\ cm^{-3}},\ 0.10)^{\rm T}.$$

The numerical solution is evolved to $t = 0.2$ ms, using 256 elements with polynomial degree $k = 2$ and SSP-RK3 time integration. To fully test the bound-enforcing limiter, we run this test without the slope limiter discussed in Section 3.3. Moreover, it is possible to design an initial state for the Sod shock-tube problem that does not place the solution close to or below the minimum table boundary. An example of this is seen in Section 4.2.1, where the bound-enforcing limiter is not required to keep the solution within the set of admissible states. However, we note that this particular test (using the initial condition described immediately above) fails without the bound-enforcing limiter, regardless of whether or not the slope limiter is implemented.[11] Thus, the bound-enforcing limiter allows for a wider selection of initial states that would otherwise cause the algorithm to fail.

Numerical results from this test are displayed in Figure 13. In the left panel, we plot the specific internal energy versus position at the end of the simulation (solid black curve). We also plot the minimum internal energy $\epsilon_{\rm min}(\rho,\, Y_{\rm e})$ (dashed red curve). Around the shock, ϵ is very close to the minimum value, as can be seen in the inset in the left panel of Figure 13. In fact, the specific internal energy remains very close to the minimum value throughout this test. The middle panel displays a spacetime plot of the limiter parameter $\vartheta_3(x,\ t)$ in Equation (74), and shows the activation sites for the bound-enforcing limiter, where the average value for ϑ_3 when limiting is required is 0.9 and it ranges from $0.60 < \vartheta_3 < 0.99$. The bound-enforcing limiter is activated due to small oscillations slightly ahead of the shock, and produces a trace of the shock trajectory as seen in the middle panel in Figure 13. The slope of the prominent trace in $\vartheta_3(x,\ t)$ indicates a shock velocity of $v_{\rm shock} \approx 1800$ km s^{-1}. Finally, the right panel in Figure 13 shows the relative change in the conserved quantities versus time. The change in these quantities are due to machine roundoff, indicating that the bound-enforcing limiter is sufficiently conservative for this test.

[11] Even when slope limiting is used, the bound-enforcing limiter is required for this test, but we decide to deactivate the slope limiter to provoke the bound-enforcing limiter even more.

4.2.4. Shu–Osher Shock Tube

This test adopted from Shu & Osher (1988) involves a Mach = 3 shock interacting with a lower-density region with a sinusoidal perturbation. As the shock propagates and interacts with the density perturbations, the perturbations move upstream, forming high-frequency oscillations just behind the shock. This problem tests the ability of a shock-capturing method to limit unphysical oscillations without destroying physical, small-scale features of the post-shock flow. We note that small-scale features resulting from hydrodynamical instabilities, such as turbulence and convection, are crucial to CCSN explosion dynamics (e.g., Murphy & Meakin 2011; Murphy et al. 2013; Couch & Ott 2015; Radice et al. 2016; Mabanta & Murphy 2018; Couch et al. 2020) and many other astrophysical applications.

Here, the problem is modified to use physical units in a regime relevant to CCSNe. The computational domain is $D = [-5,\ 5]$ km, with a discontinuity initially located at $x = 1$

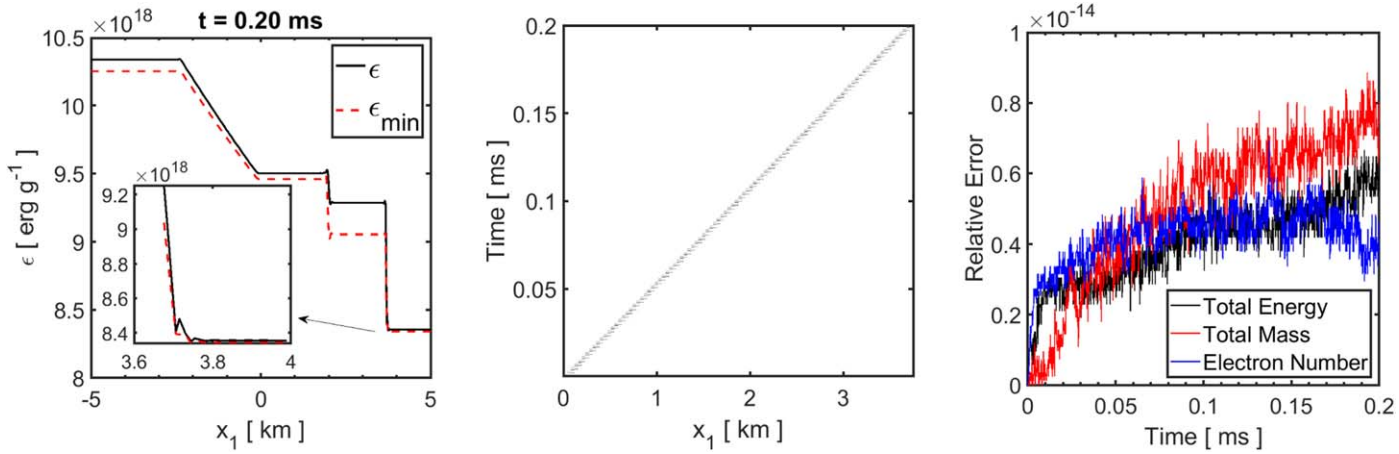

Figure 13. Numerical results for the shock tube provoking the bound-enforcing limiter. In the left panel, we plot the specific internal energy (solid black line) and the minimum specific internal energy (dashed red line) vs. position x. The middle panel shows the activation sites of the bound-enforcing limiter as indicated by a spacetime plot of $\vartheta_3(x, t)$. The solution is closest to the boundary just ahead of the shock, as indicated by the inset in the left panel. In the right panel, we plot the relative change in the conserved quantities, i.e., total fluid energy (black), mass (red), and electron number (blue).

km separating the left and right states

$$\boldsymbol{P}_{\rm L} = (\rho,\, v,\, p,\, Y_{\rm e})^{\rm T}_{\rm L} = (3.60632 \times 10^{12}\ {\rm g\ cm^{-3}},\ 7.425 \times 10^{4}\ {\rm km\ s^{-1}},\ 1.333 \times 10^{32}\ {\rm erg\ cm^{-3}},\ 0.5)^{\rm T}$$

$$\boldsymbol{P}_{\rm R} = (\rho,\, v,\, p,\, Y_{\rm e})^{\rm T}_{\rm R} = ([1 + 0.2 \times \sin(5\ {\rm km^{-1}}\, x)] \times 10^{12}\ {\rm g\ cm^{-3}},\ 0,\ 1.0 \times 10^{31}\ {\rm erg\ cm^{-3}},\ 0.5)^{\rm T}.$$

The fluid is evolved until $t = 0.0625$ ms, using 256 uniform elements and $\beta_{\rm Tvd} = 2.0$. We use third-order spatial discretization ($k = 2$) and third-order temporal integration (SSP-RK3). In this test, we compare results obtained with characteristic and componentwise limiting, and, for each limiting method, we show results for various values of the TCI threshold.

In Figure 14, we show the density obtained using characteristic (top) and componentwise (bottom) limiting for various values of the TCI threshold $C_{\rm TCI}$: 0.0 (full limiting, red), 0.03 (green), 0.3 (magenta), and 3.0 (blue), i.e., the same values that were used in Endeve et al. (2019) for the ideal EoS case. Larger values of $C_{\rm TCI}$ imply less slope limiting. These results are compared to a reference solution obtained using 2048 elements (black), with third-order spatial and temporal discretization, and $C_{\rm TCI} = 0.0$. In both limiting schemes, full limiting washes out the density variations behind the shock, while increasing the TCI threshold allows for these features to be better captured. The results obtained with $C_{\rm TCI} = 3.0$ are very close to the reference solution. However, for reasons discussed in Section 5.5, we do not recommend using such a high value for $C_{\rm TCI}$ in general, because some amount of limiting —even in smooth regions—seems to be required. For all values of the threshold (except perhaps the case with $C_{\rm TCI} = 3.0$, which applies little limiting away from the shock), the characteristic limiting scheme better captures the shape and amplitude of the oscillations behind the shock (see insets in each panel, focusing on the higher-frequency oscillations just behind the shock).

4.2.5. Two-dimensional Riemann Problem

Here we consider a 2D Riemann problem, adapted from Lax & Liu (1998), which involves a fluid with a different initial state in each quadrant given by

$$\boldsymbol{P}_{\rm NW} = (\rho,\, v^1,\, v^2,\, p,\, Y_{\rm e})^{\rm T}_{\rm NW} = (10^{12}\ {\rm g\ cm^{-3}},\ 7.275 \times 10^{4}\ {\rm km\ s^{-1}},\ 0,\ 10^{32}\ {\rm erg\ cm^{-3}},\ 0.3)^{\rm T},$$

$$\boldsymbol{P}_{\rm NE} = (\rho,\, v^1,\, v^2,\, p,\, Y_{\rm e})^{\rm T}_{\rm NE} = (5.313 \times 10^{11}\ {\rm g\ cm^{-3}},\ 0,\ 0,\ 4.0 \times 10^{31}\ {\rm erg\ cm^{-3}},\ 0.3)^{\rm T},$$

$$\boldsymbol{P}_{\rm SE} = (\rho,\, v^1,\, v^2,\, p,\, Y_{\rm e})^{\rm T}_{\rm SE} = (10^{12}\ {\rm g\ cm^{-3}},\ 0,\ 7.275 \times 10^{4}\ {\rm km\ s^{-1}},\ 10^{32}\ {\rm erg\ cm^{-3}},\ 0.3)^{\rm T},$$

$$\boldsymbol{P}_{\rm SW} = (\rho,\, v^1,\, v^2,\, p,\, Y_{\rm e})^{\rm T}_{\rm SW} = (8.0 \times 10^{11}\ {\rm g\ cm^{-3}},\ 0,\ 0,\ 10^{32}\ {\rm erg\ cm^{-3}},\ 0.3)^{\rm T}$$

on a domain $D = [0, 1.0]$ km $\times$ $[0, 1.0]$ km. This test, which corresponds to "Configuration 12" in Lax & Liu (1998), involves two shocks moving into the northeastern quadrant and contact discontinuities (or slip lines) at the northern and eastern boundaries of the southwestern quadrant. It is adapted from the original works to use physical units in a regime relevant to CCSNe with a nuclear EoS. The initial configuration presented here is one of many possible configurations of 2D Riemann problems presented in Lax & Liu (1998). The fluid is evolved until $t = 0.0025$ ms using 400^2 uniform elements, $\beta_{\rm Tvd} = 1.75$, and $C_{\rm TCI} = 0$ (i.e., limiting is applied everywhere). We use third-order spatial discretization ($k = 2$) and third-order temporal integration (SSP-RK3). To run this test, we used `thornado`'s interface to AMReX in order to take advantage of AMReX's MPI parallelization.

Figure 15 shows the density (top panels) and pressure (bottom panels) at $t = 0.0025$ ms from a run with componentwise limiting (left panels) and a run with characteristic limiting (right panels). Black lines on each plot show logarithmically spaced contours to highlight solution features. Overall, the morphology of the solutions obtained with `thornado`—using a nuclear EoS—agree well with the results displayed by Lax & Liu (1998). Moreover, the use of characteristic limiting presents a tremendous improvement over componentwise limiting, particularly as the higher dimensionality of the problem allows for more complex flow patterns and discontinuity geometries. Notably, the density and pressure contours in the componentwise limiting case reveal more oscillations and

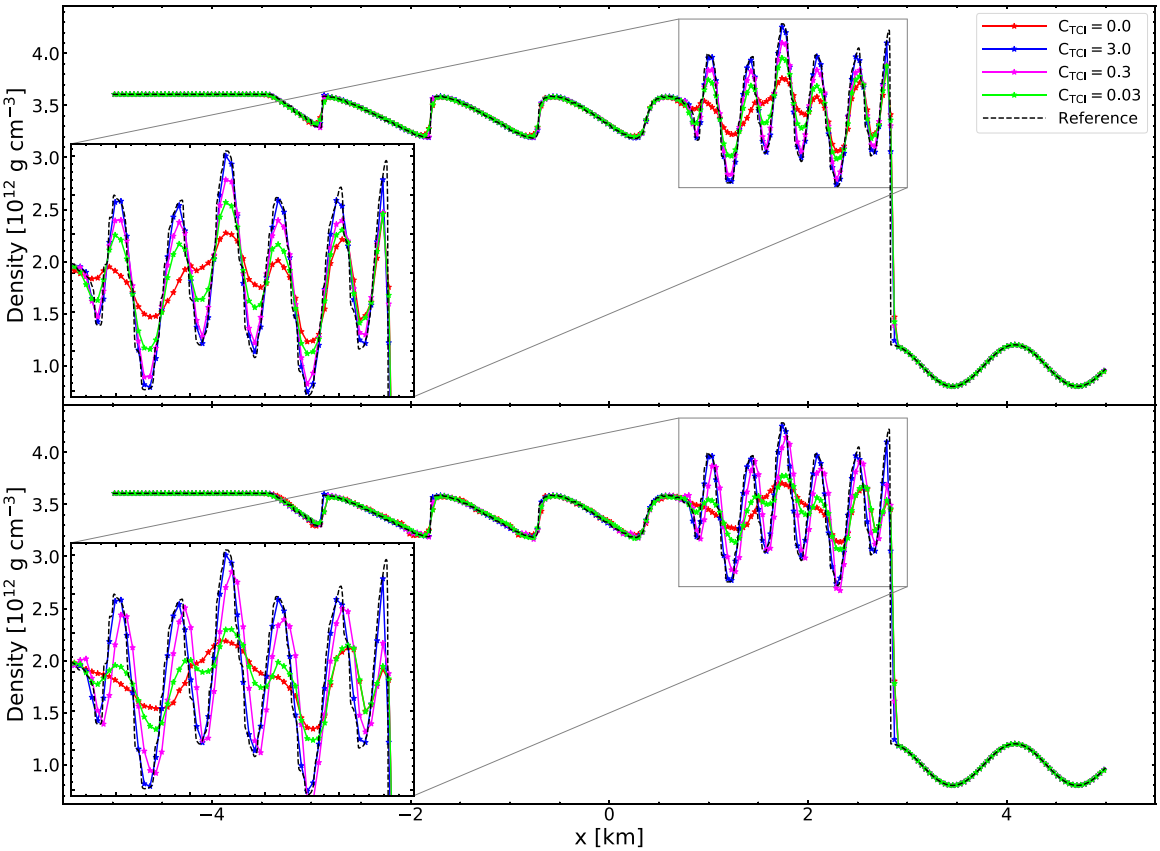

Figure 14. Numerical solution of the Shu–Osher shock tube with the nuclear EoS at $t = 0.062$ ms, using 256 elements and third-order-accurate methods with characteristic (top) and componentwise (bottom) limiting. In each panel, we plot the mass density vs. position, obtained with various values of the troubled-cell indicator threshold C_{TCI}: 0.0 (full limiting, red), 0.03 (green), 0.3 (magenta), and 3.0 (blue), compared to a reference solution (black) obtained using 2048 elements, third-order spatial discretization, and third-order time integration.

deformities. These oscillations are particularly prominent near the boundary of the curved shock surface. There appear to be no oscillations present in the run performed with characteristic limiting. Similarly, the jet-like feature seen in the southwest quadrant of the density plots appear less resolved and are somewhat asymmetric in the componentwise limiting case.

4.3. Poisson Solver Test

The accuracy of the CG method used by Poseidon to solve Equation (5) is determined by the total number of degrees of freedom used to solve the system. The number of degrees of freedom can be changed by either the *p*-method or the *h*-method. The *p*-method varies the degree k of the polynomials used in the approximation of the solution and requires $k+1$ nodes per element. The *h*-method increases the number of elements N_e used to discretize the system. These two methods are used together in the *hp*-method where both the refinement of the mesh and the degree of the approximation polynomials can be varied. In the *hp*-method, the number of degrees of freedom is given by $n_{\rm DOF} = (k+1) \times N_e$. The accuracy of the *hp*-method increases with increasing $n_{\rm DOF}$, and the error should decrease with increasing $n_{\rm DOF}$ as $1/n_{\rm DOF}^{k+1}$.

We test the accuracy of Poseidon's Poisson solver using the density profile of a centrally condensed sphere of radius R. This test, from Stone & Norman (1992), was chosen because it has a nonpolynomial analytic solution, thus allowing us to better explore the convergence properties of the solver. (Problems with polynomial solutions are solved exactly for sufficiently high k.) The density profile and analytic solution for the test are given by

$$\rho(r) = \begin{cases} \dfrac{\rho_{\rm c}}{1 + \left(\dfrac{r}{r_{\rm c}}\right)^2} & \text{if } r \leqslant R \\ 0 & \text{if } r > R \end{cases} \tag{112}$$

and

$$\Phi(r) = \begin{cases} -4\pi G \rho_{\rm c} r_{\rm c}^2 \left[1 - \dfrac{\arctan\left(\dfrac{r}{r_{\rm c}}\right)}{\dfrac{r}{r_{\rm c}}} - \dfrac{1}{2}\left(\dfrac{1 + \left(\dfrac{r}{r_{\rm c}}\right)^2}{1 + \left(\dfrac{R}{r_{\rm c}}\right)^2} \right) \right] & \text{if } r \leqslant R \\ -4\pi G \rho_{\rm c} \dfrac{r_{\rm c}^3}{r} \left[\dfrac{R}{r_{\rm c}} - \arctan\left(\dfrac{R}{r_{\rm c}}\right) \right] & \text{if } r > R, \end{cases} \tag{113}$$

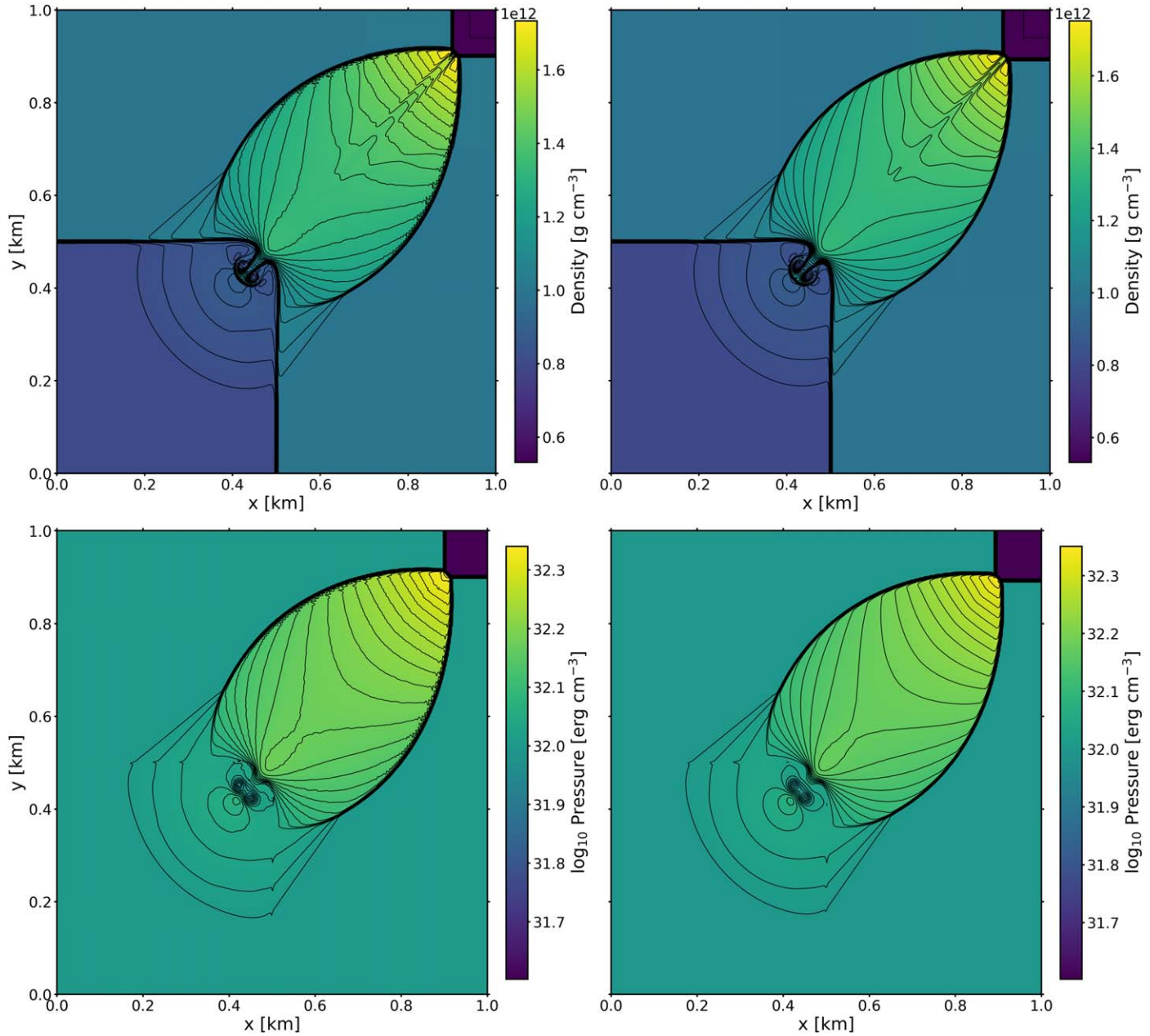

Figure 15. Numerical solution of a 2D Riemann problem (adopted from "Configuration 12" of Lax & Liu 1998) with a nuclear EoS at $t = 0.0025$ ms using 400^2 elements and third-order spatial and temporal discretization for density (top panels) and pressure (bottom panels). We compare results obtained with componentwise (left panels) and characteristic (right panels) limiting. Black lines on each plot show logarithmically spaced contours to highlight structures in the solutions.

respectively, where ρ_c and r_c are the central density and core radius, respectively. For this test, we choose $\rho_c = 150\,\mathrm{g\,cm^{-3}}$, $r_c = 0.2R_\odot$, and $R = R_\odot$, and perform the calculations over the 1D computational domain $D = [0,\ 2\,R_\odot]$. We compute the L_1 and L_∞ error norms as

$$L_1 \equiv \sum_{j=1}^{n_{\mathrm{DOF}}} |\Phi(r_j) - \Phi_h(r_j)| \tag{114}$$

and

$$L_\infty \equiv \max_j(|\Phi(r_j) - \Phi_h(r_j)|) \quad \text{for} \quad j \in \{1,\ldots,n_{\mathrm{DOF}}\}. \tag{115}$$

In Figure 16, we plot the L_1 error norm (scaled by n_{DOF}; left panel) and the L_∞ error norm (right panel) versus n_{DOF}. The numerical solutions were obtained using $k = 1$ (black symbols) and $k = 2$ (red symbols). For each value of the polynomial degree k, seven values of N_e (8, 16, 32, 64, 128, 256, and 512) were used to create uniform grids. From these plots, we see that for a specific value of N_e, the higher-order method always provides a more accurate solution. The rate of convergence observed for the third-order method is as expected (or better) in both error norms (see red, dashed reference lines). The second-order method converges at a rate somewhat slower than expected when the error is measured in the L_1 error norm, but the L_∞ error decreases roughly at the expected second-order rate (see black, dashed reference lines).

5. Adiabatic Collapse, Core Bounce, and Shock Propagation

In this section, we employ the DG method implemented in `thornado` to evolve a nonrotating progenitor through

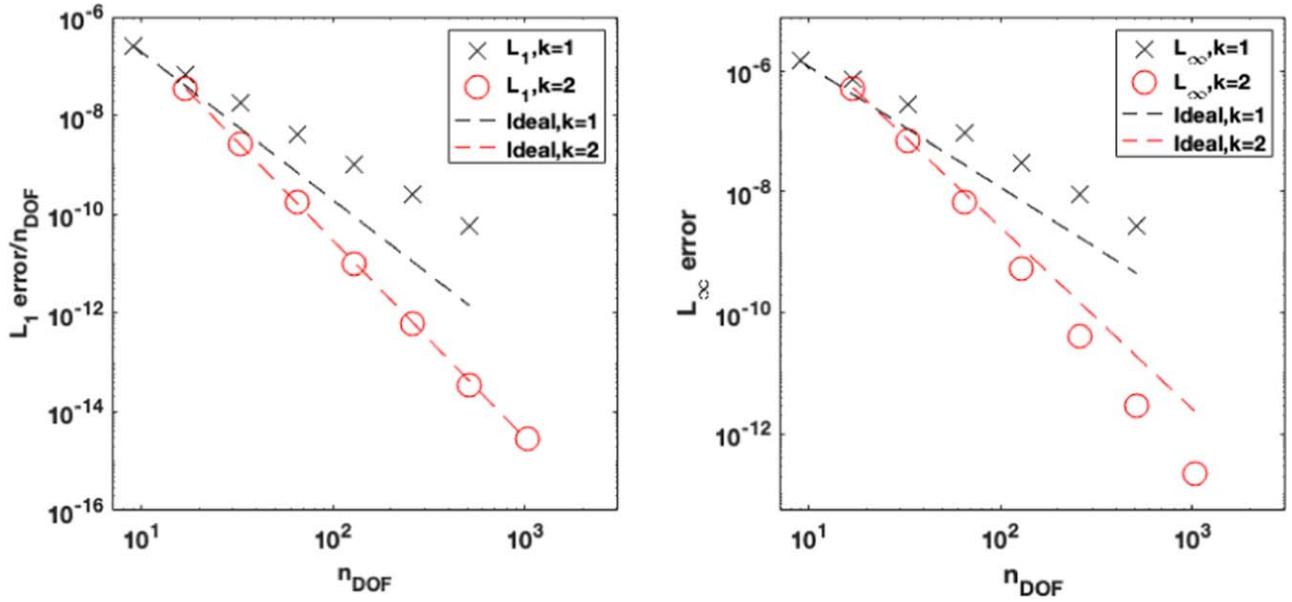

Figure 16. L_1 (left panel) and L_∞ (right panel) errors between the analytic and numerical solution calculated by the Poseidon solver for the case of a centrally condensed sphere. The L_1 error norms are scaled by the number of degrees of freedom to obtain an average error per node. The dashed lines are proportional to $1/n_{\rm DOF}^{k+1}$ and serve as references for the convergence rates of the numerical solutions.

adiabatic collapse, core bounce, and post-bounce shock propagation. The initial conditions are provided by a $15\,M_\odot$ progenitor model from Woosley & Heger (2007). Overall, this section will cover the chronological evolution of the stellar collapse model in three stages: (1) adiabatic collapse of the core, (2) core rebound and the formation of the shock shortly after nuclear saturation, and (3) the propagation of the shock through the outer core thereafter. In total, the evolution covers about 800 ms of physical time, which is divided into about 300 ms for collapse and almost 500 ms of post-bounce evolution, until the bounce shock reaches the outer boundary.

The following subsections will first discuss the physical conditions of the adiabatic collapse application that challenge any hydrodynamics method used for CCSN simulations. Then, we focus on various features of the DG method in `thornado`, such as (1) the performance of the bound-enforcing limiter during bounce and shock formation, (2) the response of the numerical solution to adjusting the TCI threshold parameter $C_{\rm TCI}$, (3) resolution dependence in the inner core, (4) the challenge of maintaining energy conservation when applying limiters, and (5) difficulties associated with employing characteristic limiting in the vicinity of the phase transition. Of course, being spherically symmetric and without neutrino transport, this adiabatic model does not describe a realistic evolutionary trajectory for a CCSN progenitor. However, this test does subject the numerical method to some of the physical conditions encountered, and we deem it a necessary step toward more realistic models.

Using spherical-polar coordinates, the domain $D = [0, 8000]$ km is divided into $N = 512$ elements. In the interest of capturing important physical characteristics while maintaining computational efficiency, this application implements a geometrically progressing grid that uses a finer spatial resolution in the inner core, which becomes progressively coarser according to

$$\Delta r_i = z \times \Delta r_{i-1}, \quad i = 2,\ldots,N, \tag{116}$$

where $z > 1$ is the "zoom factor." This emphasizes the inner core, where most of the mass is concentrated after collapse, while deemphasizing the outer regions. To begin constructing the grid, the innermost cell width $\Delta r_1 = \Delta r_{\rm min}$, the length $|D|$ of the spatial domain, and the number of elements N are defined. Then, the zoom factor is obtained by solving

$$\eta \times (z^N - 1) - (z - 1) = 0, \tag{117}$$

where $\eta = \Delta r_{\rm min}/|D|$. The fiducial run in this section uses an inner cell width of $\Delta r_{\rm min} = 0.5$ km. Then, with $|D| = 8000$ km and $N = 512$, this results in a zoom factor (in double precision) of $z = 1.009967685243838$, and an outer cell width of $\Delta r_N = 79.45$ km. Also, for the fiducial run, we use second-order spatial ($k = 1$) and temporal (SSP-RK2) discretization, combined with the componentwise limiting scheme discussed in Section 3.3, $\beta_{\rm Tvd} = 1.75$, and $C_{\rm TCI} = 0.0$. For all the runs, we use reflecting boundary conditions at the inner boundary and Dirichlet conditions (provided by the initial condition) at the outer boundary. The gravitational potential is obtained with a second-order accurate CG method as discussed in Section 3.5.

5.1. Stage 1: Collapse

Figure 17 illustrates the collapse phase prior to core bounce. We scale such that bounce occurs at $t - t_{\rm b} = 0$ ms with $t_{\rm b} = 302.9$ ms for this model, which is defined as the time when the central density, $\rho_{\rm c}$, reaches its maximum. We plot the mass density (upper-left panel), velocity (upper-right panel), electron fraction (lower-left panel), and entropy per baryon (lower-right panel) versus radius for select times during collapse. We have chosen to display the collapse profiles at the times coinciding with each full decade in central density, i.e., $\rho_{\rm c} = 10^{10,11,\ldots,14}$ g cm^{-3}. The collapse dynamics is very similar to the self-similar solutions obtained by Yahil (1983) using a polytropic EoS. The central density increases with time and approaches nuclear densities (10^{14} g cm^{-3}) at $t - t_{\rm b} = -1$ ms while the

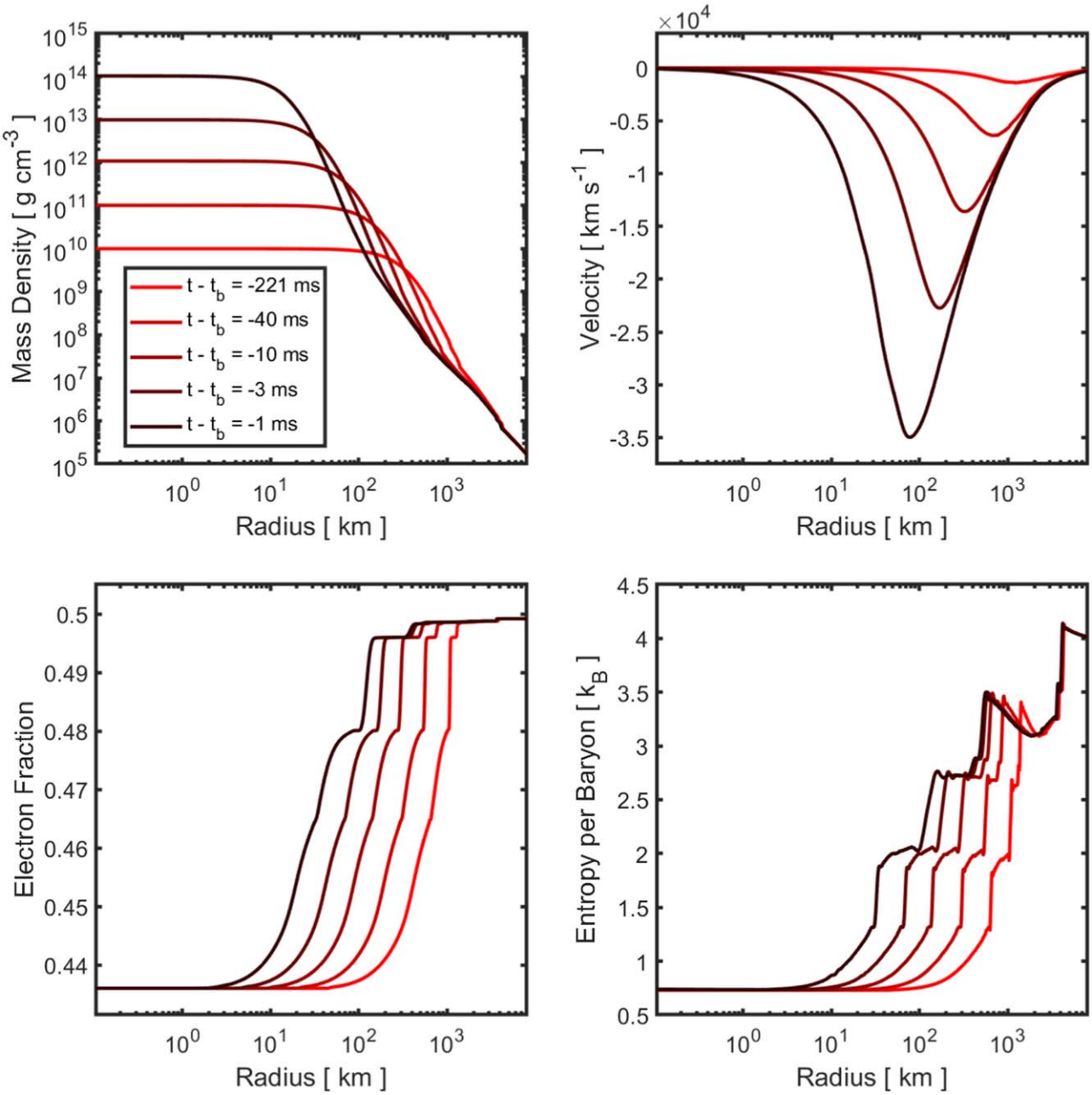

Figure 17. Numerical solutions for the adiabatic collapse of a $15M_{\odot}$ progenitor from Woosley & Heger (2007), obtained with `thornado` using 512 elements and a second-order DG scheme with componentwise limiting. Plotted vs. radius are mass density (upper left), velocity (upper right), electron fraction (lower left), and entropy per baryon (lower right) during collapse. The time slices were chosen to depict the central mass density increasing by factors of 10.

outer region rarefies as indicated by the steeper slope in density outside the innermost core. Meanwhile, the infall velocity increases linearly with radius in the inner core (consistent with homologous collapse), and approaches freefall beyond the maximum infall velocity, where it eventually falls off roughly as $r^{-1/2}$. The maximum infall velocity reaches 11%–12% of the speed of light just before bounce. The electron fraction, Y_e, is a monotonically increasing function of radius, and its inner profile shifts inward—in an approximately self-similar fashion—with the decreasing core radius during collapse. Because this test models adiabatic flows (i.e., no neutrino physics is included), the electron fraction remains constant in the core. Before core bounce and shock formation, the entropy profile shifts inward due to the collapsing core. In fact, both the electron fraction and entropy profiles remain constant in the core throughout collapse, bounce, and shock propagation, which we quantify further in Section 5.6.

5.2. Stage 2: Core Bounce

Figure 18 captures core bounce and shock formation in the inner core ($r \in [0,\ 500]$ km). We plot the adiabatic index $\Gamma \equiv \left(\frac{\partial \ln p}{\partial \ln \rho}\right)$ (upper left), velocity (upper right), electron fraction (lower left), and entropy per baryon (lower right) versus radius. In each panel, blue curves illustrate the dynamics immediately before bounce (leading up to maximum ρ_c), while red curves illustrate the dynamics immediately after bounce (see color maps to the right of each panel). The bounce

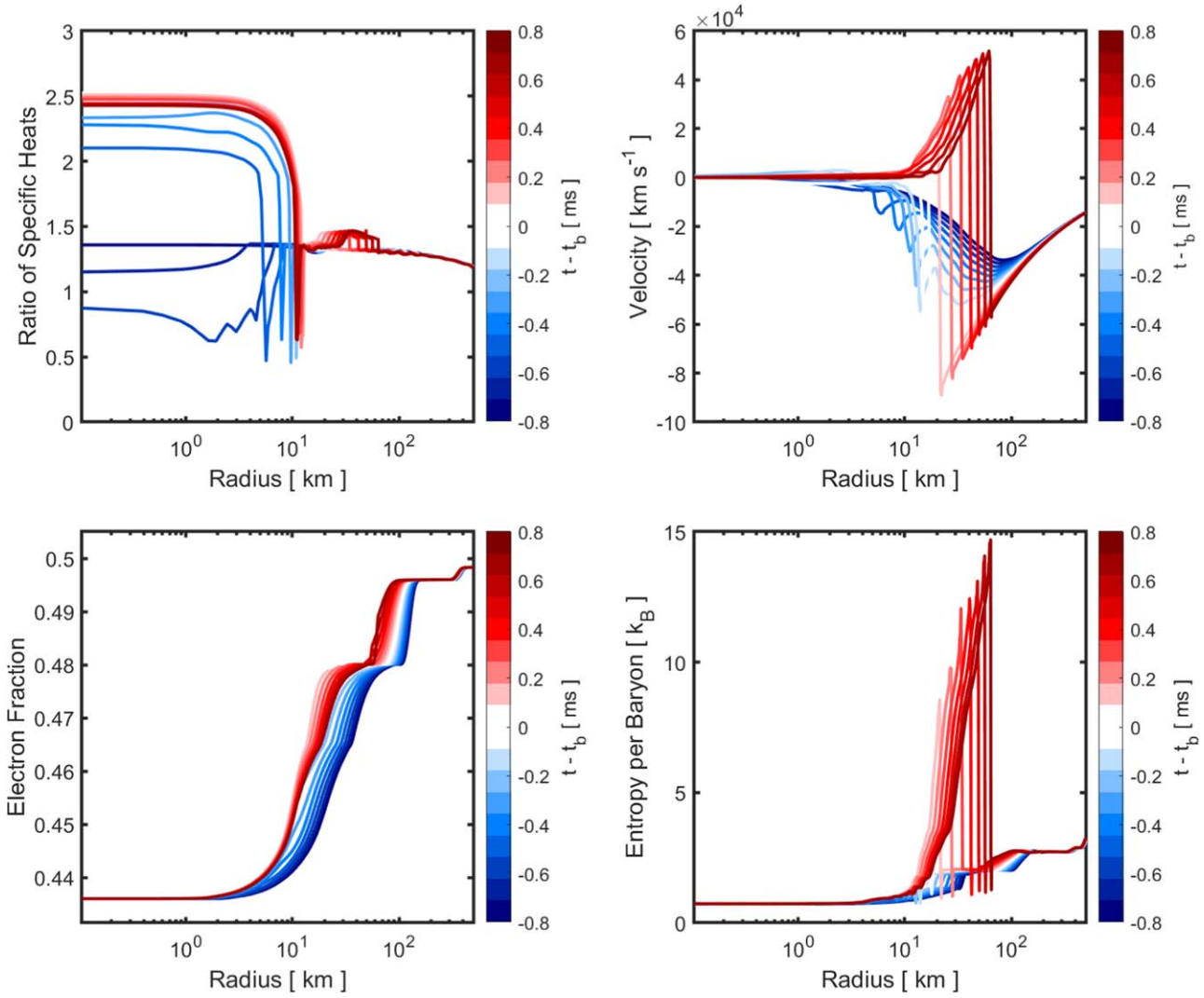

Figure 18. Numerical solutions of select quantities vs. radius for adiabatic collapse evolved through bounce: adiabatic index $\Gamma \equiv \left(\frac{\partial \ln p}{\partial \ln \rho}\right)$ (upper left), velocity (upper right), electron fraction (lower left), and entropy per baryon (lower right). A finer time resolution is used here to exhibit the characteristics of bounce and shock formation, and the color map on the right of each panel is used to distinguish pre- and post-bounce profiles: blue and red, respectively.

dynamics is in response to the stiffening of the EoS, which is illustrated by the evolution of the adiabatic index during the transition to nuclear matter in the inner core. In the upper-left panel, the adiabatic index is $\Gamma \approx 4/3$ at $t - t_b = -0.8$ ms. Once the core reaches nuclear densities and undergoes a phase transition to bulk nuclear matter, the EoS stiffens and the repulsive nuclear forces between the tightly packed nucleons result in a jump in Γ to around 2.5 at the inner boundary. After bounce, the inner core, $r \lesssim 10$ km is characterized by $\Gamma \approx 2.5$, while $\Gamma \lesssim 4/3$ at larger radii. Notice the sharp transition occurring around $r = 10$ km, which we refer to as the phase transition. The velocity profiles provide a clear demonstration of the genesis and evolution of the shock resulting immediately after bounce. When the EoS stiffens, collapse is halted, and a shockwave is formed in the region $r \in [10,\ 20]$ km. Once formed, the shock must push through the supersonically collapsing outer core. In this adiabatic simulation, without neutrinos, the shockwave propagates relatively unencumbered through the outer core, and eventually reaches the outer boundary. The constant value in electron fraction in the very inner core is preserved through bounce and shock formation, meanwhile, the profile in the outer region (around 10 km) shifts as the shock travels through. There is no noticeable change in central entropy during bounce, but, as the shock forms, there is a large increase in the entropy across the shock, as expected.

5.3. Stage 3: Shock Propagation

Figure 19 shows the shock's trajectory through the outer core on its way toward the outer boundary. In this figure, we plot the mass density (upper left), velocity (upper right), electron fraction (lower left), and temperature (lower right) versus radius for select times after bounce. As can be seen by inspecting all panels, the inner core (inside about 50 km) settles into an approximate hydrostatic equilibrium once the bounce shock has cleared. Inside this region, the velocity is small (compared with the sound speed), and the mass density, electron fraction, and temperature profiles remain practically unchanged for hundreds of milliseconds. This suggests that the DG method is quite capable of capturing the adiabatic nature of the flow (this is further supported by the results shown in the left panel in Figure 23). In the velocity figure, the shock is seen

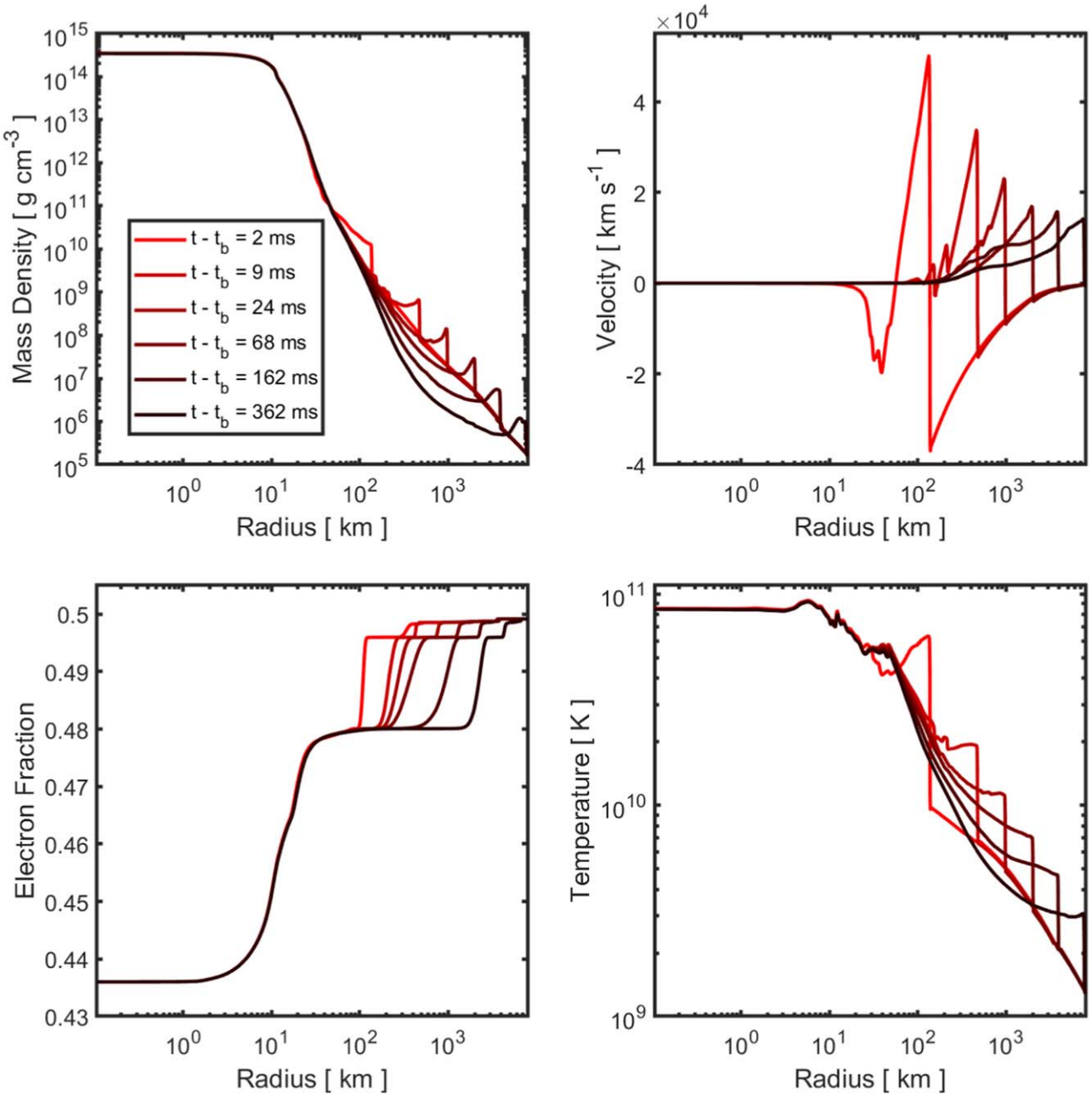

Figure 19. Numerical solutions for mass density (upper left), velocity (upper right), electron fraction (lower left), and temperature (lower right) vs. radius at select times for the adiabatic collapse simulation evolved for several hundred milliseconds post-bounce. This time domain partially captures the structure of the core as the shock propagates from its origin to the outer boundary.

to reduce in amplitude as it propagates toward the outer boundary. Early on, one can also observe secondary shocks, produced by the ring-down of the core as it settles into hydrostatic equilibrium, which later catch up with the main shock. Rarefaction of the gas occurs in the outer core (beyond 100 km) as the shock pushes through the infalling matter, notably at $t - t_b = 362$ ms in the mass density profile. The thermal energy behind the shock is partially used to dissociate heavy nuclei and alpha particles in the supersonically infalling outer core, causing the shock to lose energy while leaving behind free nucleons in its wake. As the shock travels outward, the electron fraction profile in the outer core is advected with the flow; see the sharp gradient located around 100 km at $t - t_b = 2$, which has moved to about 1000 km when $t - t_b = 362$ ms. The temperature inherently rises across the shock, and a sharp rise in temperature that traces the path of the shock is seen in the lower-right panel.

5.4. Bound-enforcing Limiter

The microphysical conditions encountered in this test are constrained by the nuclear EoS. However, some extreme conditions encountered are difficult to resolve numerically and thus may push the solutions beyond the boundaries of the admissible state set. For example, when the core bounces and launches the bounce shock, the discontinuity can generate oscillations in the numerical solution. These oscillations are to a certain degree suppressed by the slope-limiting procedure described in Section 3.3, but the solution can still exceed the

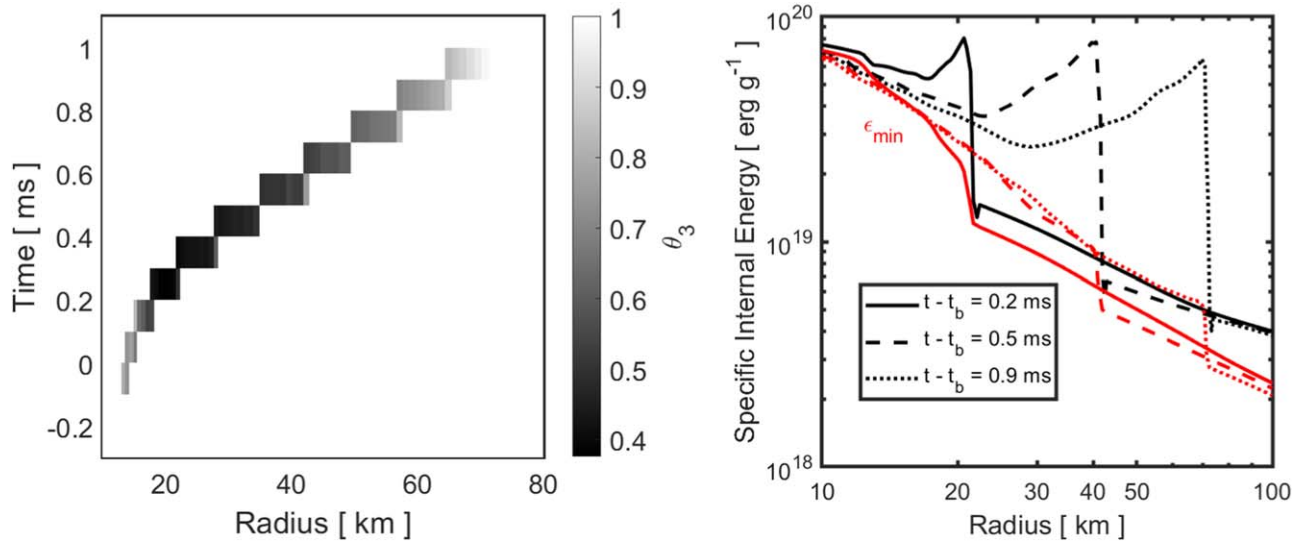

Figure 20. Activation of the bound-enforcing limiter in the fiducial adiabatic collapse simulation. The left panel shows the value of the limiter parameter ϑ_3 from Equation (74) in space and time. In the right panel, we plot the solution for ϵ (black) and the minimum $\epsilon_{\rm min}$ (red), described in Section 3.4. Each profile captures a moment in time briefly after bounce, when the bound-enforcing limiter is required to maintain $\epsilon > \epsilon_{\rm min}$.

limits of the tabulated EoS. Thus, the bound-enforcing limiting procedure from Section 3.4 is required to ensure that the numerical solution remains physically valid, mostly at bounce and shock formation. When necessary, the bound-enforcing limiter acts to constrain the mass density, electron fraction, and specific internal energy. However, for the conditions encountered in the adiabatic collapse simulations discussed in this section, only violations of the bounds on the specific internal energy trigger limiting (see Step 3 in Section 3.4), namely, during the early stages of shock formation. We note that, without the bound-enforcing limiter, the specific internal energy falls below the minimum possible value at certain locations, which then implies that a valid temperature—required, e.g., to compute the pressure—cannot be found, and the algorithm fails. Therefore, the bound-enforcing limiter is a critical component of the DG algorithm in `thornado`.

Figure 20 illustrates the action of the bound-enforcing limiter during bounce in the fiducial run discussed in the previous subsections. The left panel is a spacetime plot of the limiter parameter $\vartheta_3 \in [0,\ 1]$ (see Equation (74)), and shows the activation sites of the bound-enforcing limiter acting to constrain the specific internal energy ϵ. Values of $\vartheta_3 < 1$ imply some amount of limiting. The region displayed in the figure captures the brief moment around shock formation where ϵ drops below the minimum value but is corrected by shifting the DG solution toward the cell average by an amount determined by ϑ_3. The darker regions indicate more aggressive limiting, and we find that ϑ_3 can become as small as 0.4 in this case. In the right panel in Figure 20, the specific internal energy is plotted versus radius for select times during the initial shock propagation (black lines). We also plot the minimum specific internal energy $\epsilon_{\rm min}(\rho,\ Y_{\rm e})$, using the corresponding numerical solutions for ρ and $Y_{\rm e}$ (red lines). This figure captures ϵ being very close to, but above, $\epsilon_{\rm min}$—especially around the shock, which is located at roughly $r = 20$, 40, and 70 km for the times displayed.

Figure 21 shows the activation sites of the bound-enforcing limiter in the $\rho Y_{\rm e}$ plane (white dots). The majority of the activation sites are seen at higher mass densities and

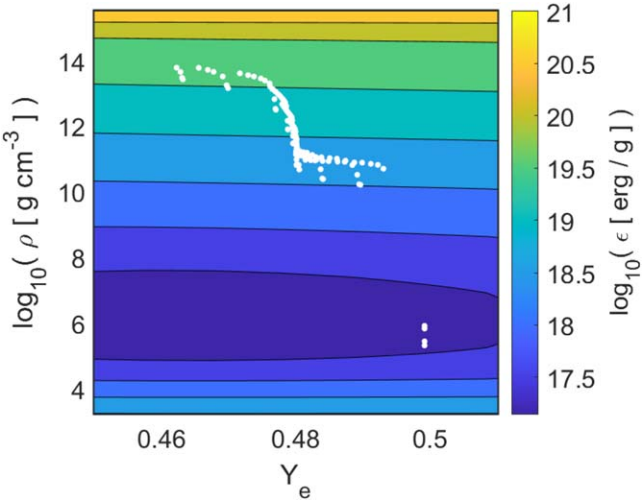

Figure 21. Activation sites (white dots) of the bound-enforcing limiter during the adiabatic collapse simulations in the $\rho Y_{\rm e}$ plane, placed over a contour plot of the surface defined by $\epsilon_{\rm min} = \epsilon\,(\rho,\ T_{\rm min},\ Y_{\rm e})$. The points in the mass density range $\log_{10}(\rho) \in [10,\ 14]$ show the limiter being activated during bounce and shock formation. The limiter is again briefly applied in the low-density region, $\log_{10}(\rho) \in [5,\ 6]$. This corresponds to when the shock momentarily forces the solution below the lower EoS table boundary as the shock passes through the outer boundary of the spatial domain.

correspond to the formation of the shock. These points appear to occupy a locally convex region of $\epsilon_{\rm min}(\rho,\ Y_{\rm e})$. However, some points also appear at a low density and higher electron fraction. These points correspond to a moment toward the end of the simulation, specifically when the shock passes through the outer boundary. This portion of the EoS table may also be locally convex; thus, the limiting scheme is expected to operate robustly in that region as well. Future work will involve an investigation of the EoS surface at minimum temperature to further challenge the robustness of our bound-enforcing limiter. This work, however, will need to be carried out in the context of neutrino radiation-hydrodynamics simulations of CCSNe, which access different and/or larger regions of the $\rho Y_{\rm e}$ plane.

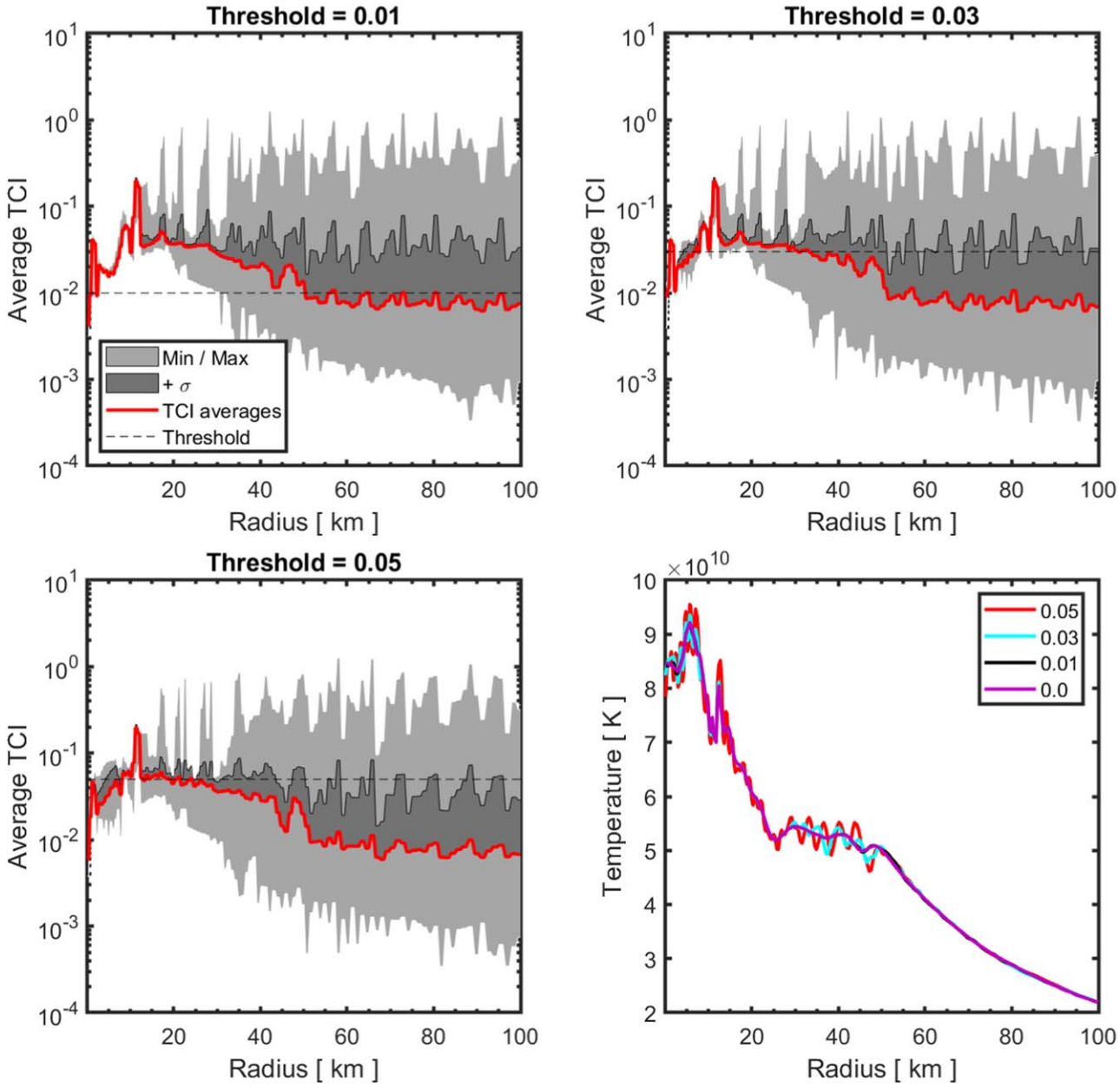

Figure 22. Troubled-cell indicator values for adiabatic collapse simulations with various thresholds $C_{\rm TCI}$ (upper panels and lower-left panel). In each panel, the red curve represents the time-averaged troubled-cell indicator value (averaged from $t_{\rm b}$ to $t_{\rm end} - t_{\rm b} = 497.1$ ms of the simulation). The lighter gray-shaded regions represent the extreme TCI values in each element (taken over all post-bounce times). The darker shaded region represents the positive TCI standard deviation. In the lower-right panel, the temperature at the end of each simulation with different $C_{\rm TCI}$ is plotted vs. radius. A higher threshold results in less slope limiting, which allows for some oscillations to develop in the temperature profile.

5.5. Troubled-cell Indicator Threshold Dependence

In this section, we investigate the effect of varying the TCI threshold $C_{\rm TCI}$ on the adiabatic collapse simulations. The numerical results discussed in the previous subsections applied the slope limiter everywhere, i.e., the TCI threshold $C_{\rm TCI}$ was set to zero such that all elements are flagged for (component-wise) limiting. As seen in Section 4.2.4, increasing the value of $C_{\rm TCI}$ prevents limiting at smooth extrema and preserves the accuracy of the solution. However, in contrast to the shock-tube problem, the solutions for the adiabatic collapse problem exhibit nonzero slopes almost everywhere. This leads to more areas that may require limiting, and it becomes more difficult to find an optimal value for $C_{\rm TCI}$. Moreover, various quantities vary by many orders of magnitude across the computational domain, and it is not clear which variables are optimal for detecting troubled cells. When using the mass density, the total fluid energy density, and the electron fraction as the variables to sense troubled cells, we find that if $C_{\rm TCI}$ is set too high, some areas that may require limiting are not flagged, and oscillations can start to develop.

In general, we have found that thermodynamic quantities such as the temperature and entropy per baryon demonstrate a higher sensitivity to $C_{\rm TCI}$ than the evolved quantities $\boldsymbol{U}$. Thus, this section will focus on the solution for the temperature and its sensitivity to $C_{\rm TCI}$. Figure 22 shows the evolution of the TCI $I_{\boldsymbol{K}}$ (see Equation (59)) versus radius for adiabatic collapse simulations with various values of $C_{\rm TCI}$: 0.01 (upper-left panel), 0.03 (upper-right panel), and 0.05 (lower-left panel). The

plotted quantities are derived from the maximum value across all fields in each element; i.e.,

$$I_{\boldsymbol{K}} = \max_{G \in \boldsymbol{G}} I_{\boldsymbol{K}}(G), \quad \text{where} \quad \boldsymbol{G} = (\rho, E, Y_e)^{\mathrm{T}}. \tag{118}$$

In each panel, the red curve represents the time-averaged value (from t_b to $t_{end} - t_b = 497.1$ ms), while the maximum and minimum values are given by the boundaries of the light gray-shaded region, and the positive standard deviations (i.e., average plus one σ) are given by the upper boundary of the dark gray-shaded region. We also plot C_{TCI} in each panel (dashed horizontal line). Recall that an element is flagged for limiting whenever $I_{\boldsymbol{K}} > C_{TCI}$. In the lower-right panel of Figure 22, we plot the temperature versus radius at the end of each simulation with a different value of C_{TCI}. For comparison, we also plot the temperature for the simulation from the previous subsections, with $C_{TCI} = 0$, which applies limiting everywhere.

As can be seen in the lower-right panel in Figure 22, the temperature profiles from all four runs display the same general trend and fall on top of each other outside $r = 50$ km. The simulations with $C_{TCI} = 0$ and $C_{TCI} = 0.01$ (magenta and black lines, respectively) are practically indistinguishable everywhere. However, inside $r = 50$ km, the simulations with the larger values of C_{TCI} (0.03 and 0.05) exhibit some oscillations about the temperature profile from the fiducial run with $C_{TCI} = 0$, and the amplitude appears to increase with increasing C_{TCI}. The TCI maxima (upper boundary of the light gray-shaded region) are generally above the threshold in all cases, which implies that limiting has been applied at least once in most of the domains displayed. However, the average and the 1σ values serve as better indicators for where limiting occurs. Although the $I_{\boldsymbol{K}}$ values tend to be above the threshold inside the first 100 km in the $C_{TCI} = 0.01$ case, the solution is limited most frequently inside $r = 50$ km, which corresponds to the region where the temperature displays oscillatory behavior in the runs with larger values of C_{TCI}. As the threshold is increased, less of this region receives limiting. And in particular, for the 0.03 and 0.05 threshold cases, oscillations have developed in this region. Ideally, the limiting procedure should both preserve the original order of accuracy and prevent the development of spurious oscillations. However, for the adiabatic collapse simulations, inside $r = 50$ km, there seems to be a trade-off between these two features, which leaves little flexibility for selecting a large value for C_{TCI}.

5.6. Resolution Dependence

In this section, we investigate the effect of varying the spatial resolution in the adiabatic collapse simulation. To do this, we keep the number of elements fixed to $N = 512$ and vary the innermost cell width Δr_1 from 0.125 to 1.0 km. Table 2 lists the inner- and outermost cell widths along with the cell widths at $r = 10$ km and $r = 100$ km, and the corresponding zoom factors z, in Equation (116). Because we keep the number of elements fixed, the zoom factor increases with decreasing Δr_1, which also results in coarser resolution in the outer regions of the computational domain. We find that the general features of the solution—e.g., density and velocity profiles—are rather insensitive to the numerical resolution. Instead, we focus on the long term evolution (i.e., hundreds of milliseconds) of the central density, electron fraction, and entropy per baryon. After bounce, when the inner core settles into hydrostatic equilibrium, the central density should remain relatively constant with time. Similarly, because we do not include neutrinos and the evolution is adiabatic, the central electron fraction and entropy per baryon should also remain constant throughout the simulation. Figure 23 shows results from varying the inner cell width. In the left panel, we plot the central density ρ_c versus time after bounce, i.e., the time when maximum central density is achieved. (To better visualize with a logarithmic abscissa, we have applied an arbitrary shift of 0.6 ms.) The right panel displays the evolution of the central entropy per baryon S_c (top) and electron fraction $Y_{e,c}$. During collapse, these quantities are plotted versus central density, while after bounce they are plotted versus time. There is some spread in the central density curves before bounce, but they all reach about the same maximum, $\rho_c \approx 4.2 \times 10^{14}$ g cm^{-3}, and, after the core stabilizes after bounce, ρ_c remains constant with time for all resolutions. For the coarsest-resolution run ($\Delta r_1 = 1$ km), the central density settles down to about 3.425×10^{14} g cm^{-3}, while in the finer-resolution models, it settles down to about 3.475×10^{14} g cm^{-3}. Because the collapse is adiabatic and the profiles are constant with radius in the very inner core (see lower panels in Figure 17), S_c and $Y_{e,c}$ should remain constant throughout the evolution. All of the simulations exhibit this behavior before $\rho_c \approx 10^{13}$ g cm^{-3}; i.e., before the phase transition into nuclear densities. (There is a slow increase in S_c, from 0.73 to 0.74, during collapse.) Just before core bounce, the profiles deviate somewhat from their constant values, and the lower resolutions exhibit larger deviations. For both central entropy and electron fraction, the profiles for the runs with $\Delta r_1 = 0.75$ km and $\Delta r_1 = 1.0$ km undergo notably larger changes than the higher-resolution profiles. $Y_{e,c}$ remains nearly constant through bounce for the 0.125, 0.25, and 0.5 km simulations. For both $Y_{e,c}$ and S_c, the two lowest resolution profiles drop further down before maximum central density, and then exhibit a slight drift with time after bounce. However, both of these quantities remain relatively constant with time after bounce in the higher-resolution cases. Thus, a threshold resolution seems to be required to accurately capture the physical behavior in the inner core. Considering the balance between computational cost and physical fidelity, an inner cell width of 0.5 km (as in the fiducial run) appears to be close to the optimal choice among the tested resolutions. For example, the central density for this run remains constant after bounce, as desired. It also maintains approximately constant central entropy and electron fraction through bounce. The central entropy deviates by no more than $0.02\,k_B$, while the electron fraction changes by no more than about 10^{-6}.

Table 2
Inner, $r = 10$ km, $r = 100$ km, Outer Cell Widths, and Zoom Factors for Geometrically Progressing Grids with $N = 512$ Elements

Δr_1 (km)	$\Delta r_{10\ \mathrm{km}}$ (km)	$\Delta r_{100\ \mathrm{km}}$ (km)	Δr_N (km)	Zoom Factor
0.125	0.258	1.430	1.048×10^2	1.013260722382225
0.25	0.366	1.401	9.225×10^1	1.011634298318296
0.5	0.598	1.489	7.945×10^1	1.009967685243838
0.75	0.835	1.630	7.185×10^1	1.008968091682754
1.0	1.077	1.821	6.641×10^1	1.008244905346311

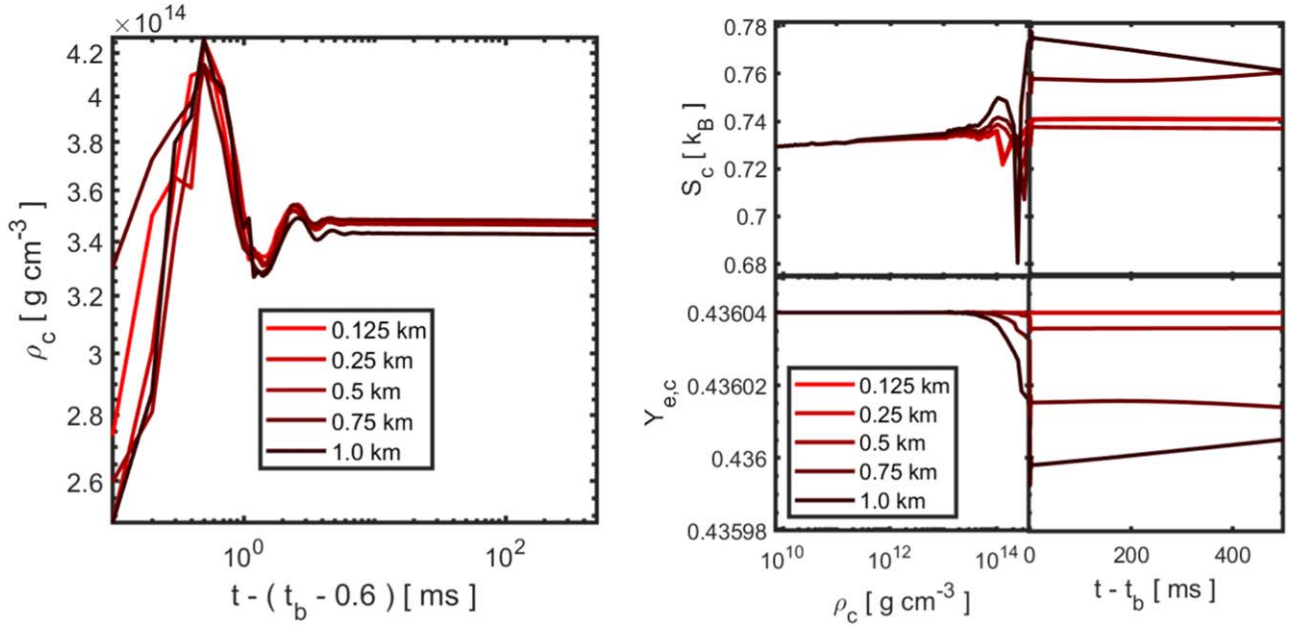

Figure 23. Results from adiabatic collapse simulations where the innermost cell width has been varied. The left panel shows the central density as a function of time for various Δr_1. The right panel shows the central entropy (top) and central electron fraction (bottom) vs. central density (up to its maximum value). Beyond the maximum central density, the entropy and electron fraction are plotted vs. time.

5.7. Energy Conservation

In this section, we investigate total energy conservation with the DG method in `thornado` in the context of adiabatic collapse simulations. The exact conservation of total energy is nontrivial to achieve in simulations of self-gravitating flows because of the adopted formulation of the fluid energy equation given by Equation (3), which is in nonconservative form due to the gravitational source term on the right-hand side. For simplicity, we limit the discussion to the present context of spherical-polar coordinates with spherical symmetry imposed. Then, by combining Equations (1), (3), and (5), it is possible to formulate a conservation law for the total energy,

$$\partial_t \mathcal{E} + \frac{1}{r^2}\partial_r(r^2\,\mathcal{F}) = 0, \tag{119}$$

where

$$\mathcal{E} = \rho\left(\epsilon + \frac{1}{2}\,v^2 + \frac{1}{2}\,\Phi\right) \quad \text{and}$$
$$\mathcal{F} = (E + p + \rho\,\Phi)v + \frac{1}{8\pi G}\left(\Phi\,\frac{\partial\dot{\Phi}}{\partial r} - \dot{\Phi}\,\frac{\partial\Phi}{\partial r}\right) \tag{120}$$

are the total energy density and total energy flux density, respectively, v is the radial component of the fluid three-velocity, and $\dot{\Phi} = \partial_t\Phi$. Because the corresponding RKDG discretization of Equations (1) and (3), and the CG discretization of Equation (5), do not combine exactly to form a discrete equivalent to Equation (119), the conservation of total energy is not expected to be exact in the adiabatic collapse simulations. Although we find that the combination of RKDG and CG discretizations exhibits surprisingly good energy conservation properties, we find evidence that the application of the slope and bound-enforcing limiters, mainly around core bounce, compromise the conservation of total energy. As seen in Figure 13 for the Riemann problem invoking the bound-enforcing limiter, in the absence of gravity, the total fluid energy (i.e., internal plus kinetic) is by construction conserved to machine precision. The slope limiter is also conservative with respect to the total fluid energy. Conservation of total energy is more difficult to achieve for self-gravitating flows such as in the adiabatic collapse problem.

By integrating Equation (119) over the computational domain $D = [0,\ R]$, and from t_0 to t, the total energy in the system is given by

$$E_{\rm total}(t) = E_{\rm total,0} - 4\pi R^2 \int_{t_0}^{t} \mathcal{F}(R,\,\tau)d\tau, \tag{121}$$

where

$$E_{\rm total} = \int_D \rho\,\epsilon\,dV + \frac{1}{2}\int_D \rho\,v^2\,dV + \frac{1}{2}\int_D \rho\,\Phi\,dV$$
$$\equiv\ E_{\rm i} + E_{\rm k} + E_{\rm g}, \tag{122}$$

and $dV = 4\pi r^2 dr$. Figure 24 shows energy conservation results from adiabatic collapse simulations. In the left panel, we plot the kinetic, gravitational, internal, and total energy versus time for the fiducial run with $\Delta r_1 = 0.5$ km. Approaching core bounce, the internal energy $E_{\rm i}$ and the gravitational energy $E_{\rm g}$ grow rapidly in concert (with opposite signs), before stabilizing after bounce with $E_{\rm i} \approx 157$ B and $E_{\rm g} \approx -158$ B, where $1\ {\rm B} = 10^{51}$ erg. The kinetic energy, $E_{\rm k}$, peaks at approximately 10 B at bounce before decreasing again and is down to 1 B when $t - t_{\rm b} = 5.5$ ms. The kinetic energy continues to decrease, and reaches a minimum of about 0.55 B at $t - t_{\rm b} \approx 40$ ms. Then, for $t - t_{\rm b} \gtrsim 40$ ms, the kinetic energy starts to increase again and is back up to 1 B for $t - t_{\rm b} = 200$ ms.

The change in the total energy versus time, $E_{\rm total} - E_{\rm total,0}$, is plotted in the middle panel of Figure 24 for the various spatial

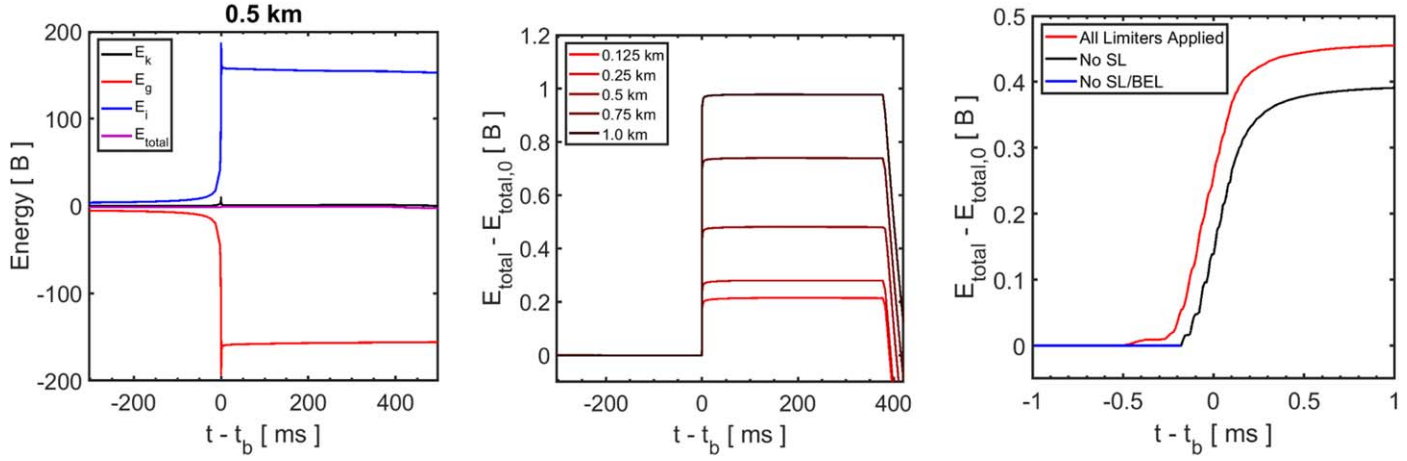

Figure 24. Energy conservation by the RKDG method in `thornado` for adiabatic collapse simulations. The left panel shows gravitational (red), kinetic (black), internal (blue), and total (magenta) energy vs. time for the fiducial run with $\Delta r_1 = 0.5$ km. Due to the relative magnitude of E_i and E_g, the details in the kinetic and total energies are obscured. The middle panel shows the change in total energy vs. time for all the resolutions considered in Section 5.6. The decrease in the total energies around $t - t_b \approx 375$ ms is due to the bounce shock reaching the outer boundary of the domain. The right panel shows the total energy vs. time for models with various combinations of limiters enabled for the fiducial run. The red line represents the total energy when applying both the slope limiter and the bound-enforcing limiter. The black line shows this quantity when only applying the bound-enforcing limiter. The blue line represents a model with both limiters off, which eventually crashed at bounce.

resolutions investigated in Section 5.6. As can be seen, the total energy remains relatively constant during collapse, makes an almost discontinuous jump around bounce, before remaining relatively constant again after bounce. (At $t - t_b \approx 375$ ms, the bounce shock reaches the outer boundary, and the total energy starts to decrease due to the energy flux through the boundary; see the second term on the right-hand side of Equation (121), which has not been accounted for in the figure.) The magnitude of the jump in total energy at bounce decreases with increasing resolution in the core. Around $t = t_b$, the total energy in the fiducial run ($\Delta r_1 = 0.5$ km) increases by less than 0.5 B, as is seen from the middle curve (after bounce) in the middle panel in Figure 24. The difference $E_{\rm total} - E_{\rm total,0}$ ought to remain zero throughout the simulation, but the extreme conditions during core bounce—due to short time and length scales, and the necessity of applying limiters around the region of shock formation, which occurs at high energy densities—result in energy conservation violations. The change in the total energy in the fiducial run is less than 0.5% of the gravitational energy at bounce and about 5% of the kinetic energy at bounce. Without accounting for the energy flowing through the outer boundary, the total energy changes by less than 1.5×10^{-3} B during collapse, until $t - t_b = -0.6$ ms, when the central density is about 1.5×10^{14} g cm^{-3}. Then, after bounce, from $t - t_b \approx 50$ ms to $t - t_b \approx 350$ ms, the total energy changes by less than 2.5×10^{-3} B, which is small compared to any of the individual components of the total energy.

We have found that the slope and bound-enforcing limiters contribute to the violation of total energy conservation at bounce. To investigate the impact of limiters on total energy conservation, we restarted the fiducial run, which employs slope and bound-enforcing limiters, at $t - t_b = -1$ ms, and ran one model with the slope limiter turned off, and one model with both the slope and bound-enforcing limiters turned off. The right panel in Figure 24 shows the total energy conservation versus time for these models. The largest violation of total energy conservation is observed in the fiducial run (red line). For the model where the slope limiter is turned off, but the bound-enforcing limiter is still active, the change in the total energy is noticeably reduced (black line). For example, the black line demonstrates no noticeable change in the total energy briefly before bounce, while the red line shows a minor increase starting at $t - t_b = -0.5$ ms. Thus, the slope limiter begins adding energy to the system shortly before bounce. Meanwhile, the bound-enforcing limiter remains inactive until about 0.2 ms before bounce. Once activated, the bound-enforcing limiter breaks total energy conservation, but to a lesser extent than when both limiters are active. The reason the limiters contribute to total energy violation is the gravitational potential energy, the third integral on the right-hand side of Equation (122). While both limiters preserve the cell-averaged fluid energy, and thus leave the first two integrals on the right-hand side of Equation (122) unchanged, the cell-averaged gravitational potential energy density is defined as a higher moment of the mass density (Φ depends on position), which is not preserved by any of the limiters. It is interesting to note that the DG method manages to model core bounce and shock formation without the slope limiter activated. When both limiters are turned off, the run fails at bounce because ϵ may fall below the minimum value required by the EoS. Until then, the DG method maintains total energy conservation well. For example, we find $E_{\rm total} - E_{\rm total}(t_b - 1\,{\rm ms}) = 8.6 \times 10^{-6}$ B at the time when the run crashes, which occurs when $\rho_c = 3.65 \times 10^{14}$ g cm^{-3}. In the future, we will investigate ways of improving the conservation of total energy while applying both limiters through bounce.

5.8. Characteristic Limiting

In contrast to the Riemann problems discussed in Section 4.2, the adiabatic collapse application does not currently benefit from characteristic limiting. As discussed earlier, toward the end of collapse, the core undergoes a phase transition from atomic nuclei and nucleons to bulk nuclear matter. However, the tabulated nuclear matter EoS appears to

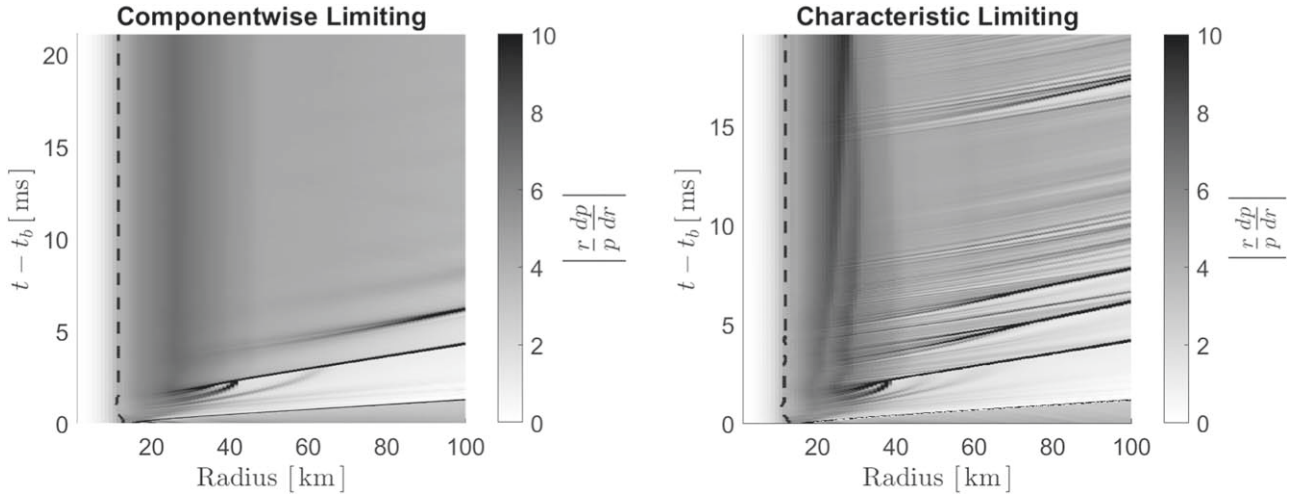

Figure 25. Spacetime plots of the absolute value of the logarithmic pressure gradient for simulations employing componentwise (left) and characteristic (right) limiting. The time domain extends over a brief period after bounce $t \in [t_b,\ t_b + 20\ \mathrm{ms}]$. The vertical dashed line around 10 km, which traces the minimum of the adiabatic index Γ_{min}, represents the approximate position of the phase transition. Near the bottom of each figure, the black line extending from approximately 20 to 100 km for a duration of 1 ms traces the bounce shock. The lines that form after this are traces of secondary or tertiary "ripples" that propagate outward from the inner core and follow the shock shortly after bounce. With componentwise limiting, the pressure gradient is relatively smooth at about 10 ms after bounce and thereafter. With characteristic limiting, perturbations continue to develop in the solution after bounce, which is visible as noise in the pressure gradient. A high time resolution was used in this case to better capture details of the dynamics around the phase transition, such as the formation of acoustic waves and their reflections off the inner boundary.

not be sufficiently smooth around this transition to enable robust construction of the characteristic fields, which depends on thermodynamic derivatives from the EoS (see Appendix B). Moreover, the interpolation scheme discussed in Section 3.6 is only C^0 continuous, which implies that derivatives are discontinuous across adjacent cubes in the table. As a result, the thermodynamic derivatives are not smooth around the phase transition, which appears to give rise to unphysical perturbations. These perturbations manifest as acoustic noise, in the characteristic and, eventually, the conserved fields, and this is clearly evident by considering the pressure. Figure 25 displays spacetime plots of the logarithmic pressure gradient, $(\ln p/\ln r)$, shortly after bounce from two simulations—one employing componentwise limiting (left panel) and one employing characteristic limiting (right panel). In both panels, the dashed black line corresponds to the minimum in the adiabatic index (see the upper-left panel in Figure 18), which we refer to as the phase transition. The formation of the bounce shock and subsequent acoustic waves during the ring-down phase after bounce are clearly seen in the lower part of both panels, which qualitatively agree. However, about 5 ms after bounce, acoustic waves are seen to continuously emanate from the core in the model with characteristic limiting. These waves are absent in the model with componentwise limiting. The acoustic waves in the model with characteristic limiting appear to emanate from the vicinity of the phase transition, i.e., originate around the vertical dashed black line (see also Figure 26). Another difference in the results obtained with the two limiters is the behavior of the maximum logarithmic pressure gradient. In the componentwise case, the peak in the pressure gradient remains fixed at approximately 30 km for the duration of the run after bounce. With characteristic limiting, this peak has a slow trajectory, starting at about $r = 15$ km and ending at $r = 30$ km.

Figure 26 shows a zoomed-in portion of the logarithmic pressure gradient for the characteristic limiting case displayed

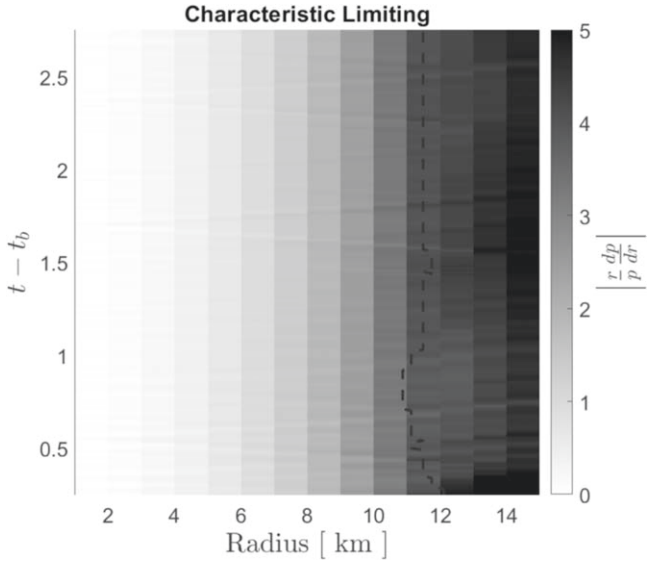

Figure 26. Zoomed-in view of the logarithmic pressure gradient (to better capture details of the dynamics around the phase transition) for the simulation employing characteristic limiting shown in the right panel in Figure 25.

in the right panel in Figure 25. Spurious pressure waves appear to be generated around the phase transition (or slightly ahead of the dashed black line), which then propagate across the entire domain. Prominent examples of this are seen around $t - t_b = 0.5$ ms, 1.65 ms, and 2.45 ms, where pairs of left- and right-propagating waves emanate from the phase transition. The left-going waves propagate toward the inner boundary and are then reflected back out. These waves lead to the noisy pattern seen in the right panel in Figure 25.

As discussed in Section 3.3, characteristic limiting relies on transforming the set of conserved variables to the set of characteristic variables by applying the matrix of left eigenvectors from the eigendecomposition of the flux Jacobian.

The construction of this matrix involves thermodynamic derivatives of the pressure and other quantities which, in the case of the tabulated EoS, do not have analytic expressions. Instead, these derivatives are obtained by differentiating the trilinear interpolation formula used to obtain quantities from the EoS table and are not necessarily smooth—especially across the phase transition. The result is a discontinuity in every characteristic variable around the location of the phase transition. Because of this, it appears to no longer be beneficial to employ characteristic limiting, as it results in the waves seen in Figures 25 and 26, and destroys the accuracy gained from characteristic limiting observed in Section 4.2. Moreover, the expression for the sound speed given in Appendix A, obtained from the eigendecomposition of the flux Jacobian, may transiently become imaginary due to variations in the derivatives. In this case, we default to constructing the sound speed as provided by the EoS table. Future work will include improving the fidelity of `thornado`'s interface with the EoS table, especially around the phase transition, in order to circumvent these problems.

6. Summary, Conclusions, and Outlook

6.1. Summary

We have extended the RKDG method for the Euler equations to accommodate an EoS for dense nuclear matter; to solve problems in Cartesian, spherical-polar, and cylindrical coordinate systems in a three-covariant framework; and to simulate adiabatic, spherically symmetric stellar collapse with self-gravity. More specifically, we have implemented a spectral-type nodal collocation DG approximation, which leads to simplifications in the semidiscrete equations—especially for problems that make use of curvilinear coordinates. In making these extensions to the RKDG method, we extended various limiters to maintain physically sound solutions:

1. We have supplemented the RKDG method with a standard TVD slope limiter, combined with a TCI, to maintain time-integration stability and to reduce spurious oscillations around discontinuities. For our purposes, this involved a nontrivial adaptation of the limiter to nuclear EoSs, specifically when limiting the characteristic fields, and we have provided the necessary characteristic decomposition to achieve this in Appendix A.
2. We have designed a bound-enforcing limiter to prevent the numerical solutions from becoming physically inadmissible, i.e., exceeding bounds imposed by the tabulated EoS. The tabulated EoS is supplied with strict boundaries in which the solution must be confined. However, critical thermodynamic quantities provided by the EoS are not necessarily globally convex, and this complicates the design of the bound-enforcing limiter, which currently operates under the assumption of a convex EoS.

We have developed `thornado` based on this extended RKDG method. `thornado` is written in modern Fortran, which is a general-purpose programming language for high-performance scientific computing. Moreover, `thornado` is intended for multiphysics CCSN simulations with high-order methods, and to this end, the RKDG method for hydrodynamics has been chosen, in part, for its ability to faithfully capture discontinuities and its ability to maintain high-order accuracy in smooth flows with a compact computational stencil. Distributed parallel computing capabilities with MPI are enabled through an interface with AMReX (Zhang et al. 2019). (The incorporation of AMReX's adaptive mesh refinement is deferred to future work.) We also mention that, in addition to distributed parallelism with MPI, `thornado` has been partially ported to utilize graphics processing units (GPUs) through the OpenACC[12] and OpenMP[13] standards, which will allow `thornado` to utilize heterogeneous architectures. Details on this progress will be reported in a future publication.

We have tested `thornado` against a suite of diverse and challenging problems incorporating a tabulated nuclear EoS in one and two spatial dimensions (see Endeve et al. 2019 for further tests in the ideal EoS case):

1. To test the formal order of accuracy of the RKDG method with a nuclear EoS, we performed an advection test with a smooth mass density profile using second- and third-order methods and various degrees of freedom to determine the rate of convergence. It was found that the third-order method is significantly more accurate than the second-order method, but the rate of convergence for the third-order method deteriorates to second order at higher resolution, possibly due to the use of trilinear EoS interpolation. To further examine the efficacy of the high-order RKDG method, a discontinuous multishaped mass density profile was advected using characteristic limiting, and the initial condition was compared with the numerical solution after 1 and 10 periods. We compared results obtained with second- and third-order methods using the same total number of degrees of freedom by adjusting the number of cells. The third-order method was found to be superior to the second-order method in this case as well.
2. We conducted several well-known Riemann problem tests—adapted to the nuclear EoS case—in Cartesian, spherical-polar, and cylindrical coordinates, and one and two spatial dimensions, to examine `thornado`'s ability to resolve discontinuities with high-order RKDG methods, without introducing spurious oscillations. It was demonstrated that results obtained with characteristic limiting are far superior to corresponding results obtained with componentwise limiting. Finally, a special version of the Sod shock-tube test was constructed to examine the efficacy of the bound-enforcing limiter. In this case, it was demonstrated that the bound-enforcing limiter maintains physically admissible solutions while at the same time preserving the inherent conservation properties of the RKDG method.

We have applied `thornado` to the problem of adiabatic stellar core collapse of a realistic nonrotating progenitor in spherical symmetry:

1. We modeled the critical phases of collapse, through nuclear densities, the phase transition to bulk nuclear matter, core bounce, shock formation, and the propagation of the shock through the outer stellar layers.
2. The complexity of this application necessitated additional investigations to probe the features of the RKDG method for hydrodynamics in `thornado`, such as the role of

[12] https://www.openacc.org
[13] https://www.openmp.org

limiting and how it contributes to improved robustness of these simulations, the dependence of the solution on the TCI threshold and spatial resolution, and the conservation of energy through challenging stages in the simulation, such as stellar core bounce.

6.2. Conclusions and Outlook

1. We successfully evolved a nonrotating, spherically symmetric, 15 $M_{\odot}$ progenitor with self-gravity through adiabatic collapse, bounce, and several hundred milliseconds of shock propagation past bounce, while maintaining adiabaticity (e.g., the entropy and electron fraction profiles remained constant in the core). The success of this application marks an important step toward applying DG methods to more realistic CCSN simulations, and given the results obtained, we are in a position to develop `thornado` further toward more physically complete CCSN simulations, e.g., by incorporating neutrino transport.
2. In the adiabatic collapse application, the bound-enforcing limiter is critical in allowing the solution to evolve through bounce. Without this limiter, the solution exceeds the limits of the EoS and the algorithm fails. The bound-enforcing limiter is required to maintain a physically valid solution for this application, but it, along with the slope limiter, interferes with the inherently good energy conservation properties of the RKDG scheme. Before and after bounce, the change in total energy is relatively low. However, when limiters are applied through bounce, an artificial jump in the total energy compromises energy conservation. The change in total energy is less than 0.5 B for the fiducial run with an inner cell width of 0.5 km and decreases with increasing spatial resolution. While the change in total energy during bounce is relatively small, when compared to any of the individual energy components, future work focusing on reducing this unphysical change in total energy is warranted.
3. For standard hydrodynamics tests with shocks, such as Riemann problems, we have shown that characteristic limiting is superior to componentwise limiting for resolving discontinuities while suppressing nonphysical, oscillatory features. However, characteristic limiting depends on derivatives of thermodynamic quantities, which are estimated from the tabulated EoS, and may not be sufficiently smooth. In particular, for the adiabatic collapse application, we observed anomalous behavior in the form of acoustic noise, which appears to originate around the phase transition. Thus, characteristic limiting currently does not provide the desired improvements for the adiabatic collapse application or any problem that may involve a phase transition. The issue associated with these thermodynamic derivatives must be further investigated and resolved before our numerical method can be extended more generally to employ more sophisticated limiters, such as moment limiters (Krivodonova 2007) or WENO-type limiters for DG (Zhu et al. 2020), which also rely on limiting characteristic fields. This may involve an improved EoS interpolation scheme that enforces thermodynamic consistency.
4. As seen in the convergence tests, the RKDG method in `thornado` gained accuracy over lower-order schemes by implementing high-order discretization ($N = 3$) for a fixed number of degrees of freedom. However, third-order methods diminished to second-order accuracy for higher degrees of freedom, and the interpolation of the tabulated EoS may have been an agent in this loss of accuracy, but further investigation is required to confirm this. Moreover, for all the tests in Section 4, our method reliably captured physical discontinuities and oscillations with high-order instantiations of the RKDG scheme in `thornado`. However, for the adiabatic collapse application, we consistently employed a second-order accurate approach. The main reason: transient spurious oscillations (or perturbations) developed when we employed third-order discretization. Again, the interpolation of the EoS may be impacting the performance of the high-order scheme. We emphasize that the results for the gravitational collapse application obtained with second-order methods and componentwise limiting are satisfactory, and provides the basis for incorporating neutrino transport algorithms also based on DG methods. However, while the present paper represents a step toward our goal, further work is required to realize CCSN simulations with high-order DG methods.
5. All results presented here were obtained with the HLL Riemann solver (Harten et al. 1983). While we have also implemented the HLLC Riemann solver (Toro et al. 1994), which is designed to account for contact discontinuities and has been shown to give superior results (see, e.g., Cardall et al. 2014), we decided not to use this Riemann solver here. The known "odd–even" instability (Quirk 1994), which develops with the HLLC Riemann solver in some multidimensional settings, is the main reason for our decision. Future work includes the development of a hybrid solver, with the capability of applying the HLLC solver in regions of smooth flow while switching to the HLL solver in the vicinity of shocks by means of a shock detector.
6. Because CCSNe are general relativistic in nature, we are extending the hydrodynamics in `thornado` to accommodate general relativity under the conformally flat approximation (see, e.g., Wilson et al. 1996), some details of which are given in Dunham et al. (2020).

E.E. and A.M., along with J.B., S.J.D., D.P., and N.R., acknowledge support from the NSF Gravitational Physics Theory Program (NSF PHY 1505933 and 1806692). B.L.B. is supported by the National Science Foundation Graduate Research Fellowship Program under grant No. DGE-1848739. E.E. was partially supported by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of the US Department of Energy Office of Science and the National Nuclear Security Administration. We thank Ann S. Almgren and Don E. Willcox for help with interfacing `thornado` with AMReX.

Software: Matplotlib[14] (Hunter 2007), NumPy[15] (Harris et al. 2020), SciPy[16] (Virtanen et al. 2020), yt (Turk et al. 2011) AMReX[17] (Zhang et al. 2019).

[14] https://matplotlib.org/
[15] http://www.numpy.org/
[16] https://www.scipy.org/
[17] https://amrex-codes.github.io/amrex/

Appendix A
Characteristic Decomposition

In this appendix, we provide the characteristic decomposition of the flux Jacobians, which are needed for slope limiting in characteristic fields. Recall that for the characteristic slope limiting described in Section 3.3, we require the eigendecomposition of the flux Jacobian,

$$\frac{\partial \boldsymbol{F}^i(\boldsymbol{U})}{\partial \boldsymbol{U}} = \mathcal{R}^i \, \Lambda^i \, (\mathcal{R}^i)^{-1} \quad (i = 1,\ldots, d). \tag{A1}$$

In the following, we will express the pressure from the EoS as $p = p(\tau, \epsilon, D_e)$, i.e., with independent variables $\tau = \rho^{-1}$, $\epsilon = e/\rho$, and $D_e = \rho\, Y_e$, instead of the usual function of ρ, T, and Y_e. This choice is arbitrary but follows the approach outlined in Colella & Glaz (1985) for a general EoS without the addition of the conservation equation for electron number (see Equation (4)). The necessary transformations of thermodynamic derivatives between these two sets of independent variables are given in Appendix B. From the state and flux vectors given in Equation (7), we can calculate the following flux Jacobian matrices in each direction:

$$\frac{\partial \boldsymbol{F}^1(\boldsymbol{U})}{\partial \boldsymbol{U}} = \begin{bmatrix} 0 & \gamma^{11} & 0 & 0 & 0 & 0 \\ -v^1 v_1 - p_\tau \tau^2 - p_\epsilon \tau\left(\epsilon - \dfrac{v^i v_i}{2}\right) & v^1(2 - p_\epsilon \tau) & -p_\epsilon v^2 \tau & -p_\epsilon v^3 \tau & p_\epsilon \tau & p_{D_e} \\ -v^1 v_2 & \gamma^{11} v_2 & v^1 & 0 & 0 & 0 \\ -v^1 v_3 & \gamma^{11} v_3 & 0 & v^1 & 0 & 0 \\ v^1\left(-H - p_\tau \tau^2 - p_\epsilon \tau\left(\epsilon - \dfrac{v^i v_i}{2}\right)\right) & \gamma^{11} H - p_\epsilon (v^1)^2 \tau & -p_\epsilon v^1 v^2 \tau & -p_\epsilon v^1 v^3 \tau & v^1(1 + p_\epsilon \tau) & v^1 p_{D_e} \\ -v^1 Y_e & \gamma^{11} Y_e & 0 & 0 & 0 & v^1 \end{bmatrix}, \tag{A2}$$

$$\frac{\partial \boldsymbol{F}^2(\boldsymbol{U})}{\partial \boldsymbol{U}} = \begin{bmatrix} 0 & 0 & \gamma^{22} & 0 & 0 & 0 \\ -v^2 v_1 & v^2 & \gamma^{22} v_1 & 0 & 0 & 0 \\ -v^2 v_2 - p_\tau \tau^2 - p_\epsilon \tau\left(\epsilon - \dfrac{v^i v_i}{2}\right) & -p_\epsilon v^1 \tau & v^2(2 - p_\epsilon \tau) & -p_\epsilon v^3 \tau & p_\epsilon \tau & p_{D_e} \\ -v^2 v_3 & 0 & \gamma^{22} v_3 & v^2 & 0 & 0 \\ v^2\left(-H - p_\tau \tau^2 - p_\epsilon \tau\left(\epsilon - \dfrac{v^i v_i}{2}\right)\right) & -p_\epsilon v^2 v^1 \tau & \gamma^{22} H - p_\epsilon (v^2)^2 \tau & -p_\epsilon v^2 v^3 \tau & v^2(1 + p_\epsilon \tau) & v^2 p_{D_e} \\ -v^2 Y_e & 0 & \gamma^{22} Y_e & 0 & 0 & v^2 \end{bmatrix}, \tag{A3}$$

and

$$\frac{\partial \boldsymbol{F}^3(\boldsymbol{U})}{\partial \boldsymbol{U}} = \begin{bmatrix} 0 & 0 & 0 & \gamma^{33} & 0 & 0 \\ -v^3 v_1 & v^3 & 0 & \gamma^{33} v_1 & 0 & 0 \\ -v^3 v_2 & 0 & v^3 & \gamma^{33} v_2 & 0 & 0 \\ -v^3 v_3 - p_\tau \tau^2 - p_\epsilon \tau\left(\epsilon - \dfrac{v^i v_i}{2}\right) & -p_\epsilon v^1 \tau & -p_\epsilon v^2 \tau & v^3(2 - p_\epsilon \tau) & p_\epsilon \tau & p_{D_e} \\ v^3\left(-H - p_\tau \tau^2 - p_\epsilon \tau\left(\epsilon - \dfrac{v^i v_i}{2}\right)\right) & -p_\epsilon v^3 v^1 \tau & -p_\epsilon v^3 v^2 \tau & \gamma^{33} H - p_\epsilon (v^3)^2 \tau & v^3(1 + p_\epsilon \tau) & v^3 p_{D_e} \\ -v^3 Y_e & 0 & 0 & \gamma^{33} Y_e & 0 & v^3 \end{bmatrix}, \tag{A4}$$

where we have defined the specific enthalpy of stagnation $H = \tau(E + p)$ and introduced the compact notation

$$p_\epsilon = \left(\frac{\partial p}{\partial \epsilon}\right)_{\tau, D_e}, \quad p_{D_e} = \left(\frac{\partial p}{\partial D_e}\right)_{\tau, \epsilon}, \quad p_\tau = \left(\frac{\partial p}{\partial \tau}\right)_{\epsilon, D_e} \tag{A5}$$

to express the necessary partial derivatives. The eigenvalues of the flux Jacobian are given by the diagonal matrix

$$\Lambda^i = \begin{bmatrix} v^i - c_s\sqrt{\gamma^{ii}} & 0 & 0 & 0 & 0 & 0 \\ 0 & v^i & 0 & 0 & 0 & 0 \\ 0 & 0 & v^i & 0 & 0 & 0 \\ 0 & 0 & 0 & v^i & 0 & 0 \\ 0 & 0 & 0 & 0 & v^i & 0 \\ 0 & 0 & 0 & 0 & 0 & v^i + c_s\sqrt{\gamma^{ii}} \end{bmatrix}, \tag{A6}$$

where $c_s = \sqrt{\Gamma p \tau}$, with

$$\Gamma = (\tau(p p_\epsilon - p_\tau) + p_{D_e} Y_e \tau^{-1}) p^{-1}, \tag{A7}$$

is the local sound speed. In the less general case where we ignore the electron contribution (i.e., $p_{D_e} = 0$), this reduces to the expression given by Colella & Glaz (1985). The right eigenvectors are then given by the column vectors of the following matrices:

$$\mathcal{R}^1 = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 1 \\ v_1 - c_s\sqrt{\gamma_{11}} & 0 & v_1 & v_1 & 0 & v_1 + c_s\sqrt{\gamma_{11}} \\ v_2 & 1 & 0 & 0 & 0 & v_2 \\ v_3 & 0 & 0 & 0 & 1 & v_3 \\ H - c_s\sqrt{\gamma_{11}} v^1 & v^2 & \beta_1 & 0 & v^3 & H + c_s\sqrt{\gamma_{11}} v^1 \\ Y_e & 0 & 0 & \dfrac{\tau\chi_1}{2p_{D_e}} & 0 & Y_e \end{bmatrix}, \tag{A8}$$

$$\mathcal{R}^2 = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 1 \\ v_1 & 1 & 0 & 0 & 0 & v_1 \\ v_2 - c_s\sqrt{\gamma_{22}} & 0 & v_2 & v_2 & 0 & v_2 + c_s\sqrt{\gamma_{22}} \\ v_3 & 0 & 0 & 0 & 1 & v_3 \\ H - c_s\sqrt{\gamma_{22}} v^2 & v^1 & \beta_2 & 0 & v^3 & H + c_s\sqrt{\gamma_{22}} v^2 \\ Y_e & 0 & 0 & \dfrac{\tau\chi_2}{2p_{D_e}} & 0 & Y_e \end{bmatrix}, \tag{A9}$$

and

$$\mathcal{R}^3 = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 1 \\ v_1 & 1 & 0 & 0 & 0 & v_1 \\ v_2 & 0 & 0 & 0 & 1 & v_2 \\ v_3 - c_s\sqrt{\gamma_{33}} & 0 & v_3 & v_3 & 0 & v_3 + c_s\sqrt{\gamma_{33}} \\ H - c_s\sqrt{\gamma_{33}} v^3 & v^1 & \beta_3 & 0 & v^2 & H + c_s\sqrt{\gamma_{33}} v^3 \\ Y_e & 0 & 0 & \dfrac{\tau\chi_3}{2p_{D_e}} & 0 & Y_e \end{bmatrix}, \tag{A10}$$

where the following definitions have been used:

$$\begin{aligned} \Delta_1 &= 2v^1 v_1 - v^i v_i, \\ \Delta_2 &= 2v^2 v_2 - v^i v_i, \\ \Delta_3 &= 2v^3 v_3 - v^i v_i, \\ \chi_i &= p_\epsilon(\Delta_i + 2\epsilon) + 2p_\tau\tau, \\ \beta_i &= \frac{1}{2}\left(\Delta_i + 2\epsilon + \frac{2p_\tau\tau}{p_\epsilon}\right). \end{aligned}$$

The left eigenvectors are given by the row vectors of the inverse matrix $\mathcal{L}^i = (\mathcal{R}^i)^{-1}$,

$$(\mathcal{R}^1)^{-1} = \frac{1}{c_s^2}\begin{bmatrix} \frac{1}{4}(\omega + 2c_s\sqrt{\gamma_{11}}v^1) & -\frac{1}{2}(c_s\sqrt{\gamma^{11}} + \phi^1) & -\frac{1}{2}\phi^2 & -\frac{1}{2}\phi^3 & \frac{p_\epsilon \tau}{2} & \frac{p_{D_e}}{2} \\ -\frac{v_2\omega}{2} & \phi^1 v_2 & c_s^2 + \phi^2 v_2 & \phi^3 v_2 & -\phi_2 & -p_{D_e} v_2 \\ \frac{2\chi_1 c_s^2 + \alpha_1\omega\tau^{-1}}{2\chi_1} & -\frac{\phi^1\alpha_1}{\chi_1\tau} & -\frac{\phi^2\alpha_1}{\chi_1\tau} & -\frac{\phi^3\alpha_1}{\chi_1\tau} & \frac{p_\epsilon\alpha_1}{\chi_1} & \frac{p_{D_e}(\alpha_1 - 2c_s^2)}{\tau\chi_1} \\ -\frac{Y_e p_{D_e}\omega}{\chi_1\tau} & \frac{2Y_e p_{D_e}\phi^1}{\chi_1\tau} & \frac{2Y_e p_{D_e}\phi^2}{\chi_1\tau} & \frac{2Y_e p_{D_e}\phi^3}{\chi_1\tau} & -\frac{2Y_e p_{D_e} p_\epsilon}{\chi_1} & \frac{2p_{D_e}(c_s^2 - Y_e p_{D_e})}{\tau\chi_1} \\ -\frac{v_3\omega}{2} & \phi^1 v_3 & \phi^2 v_3 & c_s^2 + \phi^3 v_3 & -\phi_3 & -p_{D_e} v_3 \\ \frac{1}{4}(\omega - 2c_s\sqrt{\gamma_{11}}v^1) & \frac{1}{2}(c_s\sqrt{\gamma^{11}} - \phi^1) & -\frac{1}{2}\phi^2 & -\frac{1}{2}\phi^3 & \frac{p_\epsilon\tau}{2} & \frac{p_{D_e}}{2} \end{bmatrix}, \tag{A11}$$

$$(\mathcal{R}^2)^{-1} = \frac{1}{c_s^2}\begin{bmatrix} \frac{1}{4}(\omega + 2c_s\sqrt{\gamma_{22}}v^2) & -\frac{1}{2}\phi^1 & -\frac{1}{2}(c_s\sqrt{\gamma^{22}} + \phi^2) & -\frac{1}{2}\phi^3 & \frac{p_\epsilon\tau}{2} & \frac{p_{D_e}}{2} \\ -\frac{v_1\omega}{2} & c_s^2 + \phi^1 v_1 & \phi^2 v_1 & \phi^3 v_1 & -\phi_1 & -p_{D_e} v_1 \\ \frac{2\chi_2 c_s^2 + \alpha_2\omega\tau^{-1}}{2\chi_2} & -\frac{\phi^1\alpha_2}{\chi_2\tau} & -\frac{\phi^2\alpha_2}{\chi_2\tau} & -\frac{\phi^3\alpha_2}{\chi_2\tau} & \frac{p_\epsilon\alpha_2}{\chi_2} & \frac{p_{D_e}(\alpha_2 - 2c_s^2)}{\tau\chi_2} \\ -\frac{Y_e p_{D_e}\omega}{\chi_2\tau} & \frac{2Y_e p_{D_e}\phi^1}{\chi_2\tau} & \frac{2Y_e p_{D_e}\phi^2}{\chi_2\tau} & \frac{2Y_e p_{D_e}\phi^3}{\chi_2\tau} & -\frac{2Y_e p_{D_e} p_\epsilon}{\chi_2} & \frac{2p_{D_e}(c_s^2 - Y_e p_{D_e})}{\tau\chi_2} \\ -\frac{v_3\omega}{2} & \phi^1 v_3 & \phi^2 v_3 & c_s^2 + \phi^3 v_3 & -\phi_3 & -p_{D_e} v_3 \\ \frac{1}{4}(\omega - 2c_s\sqrt{\gamma_{22}}v^2) & -\frac{1}{2}\phi^1 & \frac{1}{2}(c_s\sqrt{\gamma^{22}} - \phi^2) & -\frac{1}{2}\phi^3 & \frac{p_\epsilon\tau}{2} & \frac{p_{D_e}}{2} \end{bmatrix}, \tag{A12}$$

and

$$(\mathcal{R}^3)^{-1} = \frac{1}{c_s^2}\begin{bmatrix} \frac{1}{4}(\omega + 2c_s\sqrt{\gamma_{33}}v^3) & -\frac{1}{2}\phi^1 & -\frac{1}{2}\phi^2 & -\frac{1}{2}(c_s\sqrt{\gamma^{33}} + \phi^3) & \frac{p_\epsilon\tau}{2} & \frac{p_{D_e}}{2} \\ -\frac{v_1\omega}{2} & c_s^2 + \phi^1 v_1 & \phi^2 v_1 & \phi^3 v_1 & -\phi_1 & -p_{D_e} v_1 \\ \frac{2\chi_3 c_s^2 + \alpha_3\omega\tau^{-1}}{2\chi_3} & -\frac{\phi^1\alpha_3}{\chi_3\tau} & -\frac{\phi^2\alpha_3}{\chi_3\tau} & -\frac{\phi^3\alpha_3}{\chi_3\tau} & \frac{p_\epsilon\alpha_3}{\chi_3} & \frac{p_{D_e}(\alpha_3 - 2c_s^2)}{\tau\chi_3} \\ -\frac{Y_e p_{D_e}\omega}{\chi_3\tau} & \frac{2Y_e p_{D_e}\phi^1}{\chi_3\tau} & \frac{2Y_e p_{D_e}\phi^2}{\chi_3\tau} & \frac{2Y_e p_{D_e}\phi^3}{\chi_3\tau} & -\frac{2Y_e p_{D_e} p_\epsilon}{\chi_3} & \frac{2p_{D_e}(c_s^2 - Y_e p_{D_e})}{\tau\chi_3} \\ -\frac{v_2\omega}{2} & \phi^1 v_2 & c_s^2 + \phi^2 v_2 & \phi^3 v_2 & -\phi_2 & -p_{D_e} v_2 \\ \frac{1}{4}(\omega - 2c_s\sqrt{\gamma_{33}}v^3) & -\frac{1}{2}\phi^1 & -\frac{1}{2}\phi^2 & \frac{1}{2}(c_s\sqrt{\gamma^{33}} - \phi^3) & \frac{p_\epsilon\tau}{2} & \frac{p_{D_e}}{2} \end{bmatrix}, \tag{A13}$$

where $\phi_i = p_\epsilon \tau v_i$, $\phi^i = p_\epsilon \tau v^i$, $\omega = \tau(p_\epsilon(v^i v_i - 2\epsilon) - 2p_\tau\tau)$, and $\alpha_i = 2Y_e p_{D_e} - \tau\chi_i$.

Appendix B
Thermodynamic Derivatives

In Appendix A, to compute the flux Jacobian matrices, we expressed the pressure $p = p(\tau, \epsilon, D_e)$ in terms of the independent variables $\tau = \rho^{-1}$, $\epsilon = e/\rho$, and $D_e = \rho Y_e$. On the other hand, the tabulated EoS constructs thermodynamic variables in terms of ρ, T, and Y_e. Thus, we need to express the thermodynamic derivatives of pressure necessary for the characteristic decomposition in terms of the independent variables from the EoS table. We start with the differential of pressure,

$$dp = (\partial_\tau p)_{\epsilon,D_e} d\tau + (\partial_\epsilon p)_{\tau,D_e} d\epsilon + (\partial_{D_e} p)_{\epsilon,\tau} dD_e. \tag{B1}$$

Similarly, we may express the differentials of τ, ϵ, and D_e in terms of differentials of the table variables

$$\begin{aligned} d\tau &= -\rho^{-2} d\rho, \\ d\epsilon &= (\partial_\rho \epsilon)_{T,Y_e} d\rho + (\partial_T \epsilon)_{\rho,Y_e} dT + (\partial_{Y_e} \epsilon)_{T,\rho} dY_e, \\ dD_e &= (\partial_\rho D_e)_{T,Y_e} d\rho + (\partial_T D_e)_{\epsilon,Y_e} dT + (\partial_{Y_e} D_e)_{T,\epsilon} dY_e \\ &= Y_e d\rho + \rho dY_e. \end{aligned} \tag{B2}$$

Inserting these differentials into Equation (B1), we find another expression for the pressure differential,

$$\begin{aligned} dp = &[-\rho^2 (\partial_\tau p)_{\epsilon,D_e} + (\partial_\rho \epsilon)_{T,Y_e} (\partial_\epsilon p)_{\tau,D_e} + Y_e (\partial_{D_e} p)_{\epsilon,\tau}] d\rho \\ &+ (\partial_T \epsilon)_{\rho,Y_e} (\partial_\epsilon p)_{\tau,D_e} dT + [(\partial_{Y_e} \epsilon)_{T,\rho} (\partial_\epsilon p)_{\tau,D_e} + \rho (\partial_{D_e} p)_{\epsilon,\tau}] dY_e. \end{aligned} \tag{B3}$$

On the other hand, we have the differential of pressure in terms of the table variables,

$$dp = (\partial_\rho p)_{T,Y_e} d\rho + (\partial_T p)_{\rho,Y_e} dT + (\partial_{Y_e} p)_{T,\rho} dY_e. \tag{B4}$$

Comparing Equations (B3) and (B4), we have the system of equations

$$\begin{bmatrix} -\rho^2 & (\partial_\rho \epsilon)_{T,Y_e} & Y_e \\ 0 & (\partial_T \epsilon)_{\rho,Y_e} & 0 \\ 0 & (\partial_{Y_e} \epsilon)_{T,\rho} & \rho \end{bmatrix} \begin{bmatrix} (\partial_\tau p)_{\epsilon,D_e} \\ (\partial_\epsilon p)_{\tau,D_e} \\ (\partial_{D_e} p)_{\epsilon,\tau} \end{bmatrix} = \begin{bmatrix} (\partial_\rho p)_{T,Y_e} \\ (\partial_T p)_{\rho,Y_e} \\ (\partial_{Y_e} p)_{T,\rho} \end{bmatrix}. \tag{B5}$$

Solving, with some simplifications, we find the derivatives of the pressure with respect to τ, ϵ, and D_e in terms of the table variables ρ, T, and Y_e:

$$\left(\frac{\partial p}{\partial \epsilon}\right)_{\tau,D_e} = \left(\frac{\partial \epsilon}{\partial T}\right)^{-1}_{\rho,Y_e} \left(\frac{\partial p}{\partial T}\right)_{\rho,Y_e}, \tag{B6}$$

$$\left(\frac{\partial p}{\partial D_e}\right)_{\tau,\epsilon} = \tau \left[\left(\frac{\partial p}{\partial Y_e}\right)_{\rho,T} - \left(\frac{\partial \epsilon}{\partial Y_e}\right)_{\rho,T} \left(\frac{\partial p}{\partial \epsilon}\right)_{\tau,D_e}\right], \tag{B7}$$

$$\left(\frac{\partial p}{\partial \tau}\right)_{\epsilon,D_e} = \tau^{-2} \left[Y_e \left(\frac{\partial p}{\partial D_e}\right)_{\tau,\epsilon} + \left(\frac{\partial \epsilon}{\partial \rho}\right)_{Y_e,T} \left(\frac{\partial p}{\partial \epsilon}\right)_{\tau,D_e} - \left(\frac{\partial p}{\partial \rho}\right)_{T,Y_e}\right]. \tag{B8}$$

We use these relations to relate derivatives needed for the characteristic decomposition in Appendix A to derivatives obtained from table interpolations.

ORCID iDs

David Pochik https://orcid.org/0000-0002-8310-0271
Brandon L. Barker https://orcid.org/0000-0002-8825-0893
Eirik Endeve https://orcid.org/0000-0003-1251-9507
Jesse Buffaloe https://orcid.org/0000-0002-7712-5835
Samuel J. Dunham https://orcid.org/0000-0003-4008-6438
Nick Roberts https://orcid.org/0000-0003-1891-8402
Anthony Mezzacappa https://orcid.org/0000-0001-9816-9741

References

Abbott, B. P., Abbott, R., Abbott, T. D., et al. 2016, PhRvL, 116, 061102
Abbott, B. P., Abbott, R., Abbott, T. D., et al. 2017a, PhRvL, 118, 221101
Abbott, B. P., Abbott, R., Abbott, T. D., et al. 2017b, PhRvL, 119, 161101
Abbott, R., Abbott, T. D., Abraham, S., et al. 2020, ApJL, 896, L44
Abdikamalov, E., Ott, C. D., Radice, D., et al. 2015, ApJ, 808, 70
Adams, M. L. 2001, NSE, 137, 298
Akiyama, S., Wheeler, J. C., Meier, D. L., & Lichtenstadt, I. 2003, ApJ, 584, 954
Almgren, A. S., Beckner, V. E., Bell, J. B., et al. 2010, ApJ, 715, 1221
Arnett, W. D., & Meakin, C. 2011, ApJ, 733, 78
Baron, E., & Cooperstein, J. 1990, ApJ, 353, 597
Bassi, F., Franchina, N., Ghidoni, A., & Rebay, S. 2013, IJNMF, 71, 1322
Bethe, H. A. 1990, RvMP, 62, 801
Bethe, H. A., & Wilson, J. R. 1985, ApJ, 295, 14
Biswas, R., Devine, K. D., & Flaherty, J. E. 1994, ApNM, 14, 255
Blondin, J., & Lufkin, E. 1993, ApJS, 88, 589
Blondin, J. M., Mezzacappa, A., & DeMarino, C. 2003, ApJ, 584, 971
Bruenn, S. W. 1985, ApJS, 58, 771
Bruenn, S. W., Blondin, J. M., Hix, W. R., et al. 2020, ApJS, 248, 11
Bruenn, S. W., Raley, E. A., & Mezzacappa, A. 2004, arXiv:astro-ph/0404099
Buras, R., Janka, H. T., Rampp, M., & Kifonidis, K. 2006, A&A, 457, 281
Burrows, A. 2013, RvMP, 85, 245
Cardall, C. Y., Budiardja, R. D., Endeve, E., & Mezzacappa, A. 2014, ApJS, 210, 17
Casanova, J., Endeve, E., Lentz, E. J., et al. 2020, PhyS, 95, 064005
Chu, R., Endeve, E., Hauck, C. D., & Mezzacappa, A. 2019, JCoPh, 389, 62
Cockburn, B., Hou, S., & Shu, C.-W. 1990, MaCom, 54, 545
Cockburn, B., Lin, S.-Y., & Shu, C.-W. 1989, JCoPh, 84, 90
Cockburn, B., & Shu, C.-W. 1989, MaCom, 52, 411
Cockburn, B., & Shu, C.-W. 1991, MMNA, 25, 337
Cockburn, B., & Shu, C.-W. 1998, JCoPh, 141, 199
Cockburn, B., & Shu, C.-W. 2001, JSCom, 16, 173
Colella, P., & Glaz, H. M. 1985, JCoPh, 59, 264
Couch, S. M. 2017, RSPTA, 375, 20160271
Couch, S. M., & Ott, C. D. 2013, ApJL, 778, L7
Couch, S. M., & Ott, C. D. 2015, ApJ, 799, 5
Couch, S. M., Warren, M. L., & O'Connor, E. P. 2020, ApJ, 890, 127
Davies, S. F. 1988, SIAM J. Sci. Stat. Comput., 9, 445
Dubey, A., Antypas, K., Ganapathy, M. K., et al. 2009, ParC, 35, 512
Dumbser, M., Zanotti, O., Loubère, R., & Diot, S. 2014, JCoPh, 278, 47
Dunham, S. J., Endeve, E., Mezzacappa, A., Buffaloe, J., & Holley-Bockelmann, K. 2020, JPhCS, 1623, 012012
Endeve, E., Buffaloe, J., Dunham, S. J., et al. 2019, JPhCS, 1225, 012014

Endeve, E., Cardall, C. Y., Budiardja, R. D., et al. 2012, ApJ, 751, 26
Endeve, E., Hauck, C. D., Xing, Y., & Mezzacappa, A. 2015, JCoPh, 287, 151
Fambri, F., Dumbser, M., Köppel, S., Rezzolla, L., & Zanotti, O. 2018, MNRAS, 477, 4543
Fryxell, B., Olson, K., Ricker, P., et al. 2000, ApJS, 131, 273
Fu, G., & Shu, C.-W. 2017, JCoPh, 347, 305
Gottlieb, E., Shu, C.-W., & Tadmor, E. 2001, SIAMR, 43, 89
Greif, S. K., Hebeler, K., Lattimer, J. M., Pethick, C. J., & Schwenk, A. 2020, ApJ, 901, 155
Hanke, F., Müller, B., Wongwathanarat, A., Marek, A., & Janka, H.-T. 2013, ApJ, 770, 66
Harris, C. R., Jarrod Millman, K., van der Walt, S. J., et al. 2020, Natur, 585, 357
Harten, A., Lax, P., & van Leer, B. 1983, SIAMR, 1983, 35
Heger, A., Woosley, S. E., & Spruit, H. C. 2005, ApJ, 626, 350
Hempel, M., Fischer, T., Schaffner-Bielich, J., & Liebendörfer, M. 2012, ApJ, 748, 70
Herant, M., Benz, W., & Colgate, S. 1992, ApJ, 395, 642
Hesthaven, J. S., & Warburton, T. 2008, Nodal Discontinuous Galerkin Methods: Algorithms, Analysis and Applications (Berlin: Springer)
Hix, W. R., Lentz, E. J., Endeve, E., et al. 2014, AIPA, 4, 041013
Hunter, J. D. 2007, CSE, 9, 90
Janka, H.-T., Hanke, F., Hüdepohl, L., et al. 2012, PTEP, 2012, 01A309
Janka, H.-T., Melson, T., & Summa, A. 2016, ARNPS, 66, 341
Just, O., Obergaulinger, M., & Janka, H. T. 2015, MNRAS, 453, 3386
Käppeli, R., & Mishra, S. 2016, A&A, 587, A94
Kasen, D., Metzger, B., Barnes, J., Quataert, E., & Ramirez-Ruiz, E. 2017, Natur, 551, 80
Kidder, L. E., Field, S. E., Foucart, F., et al. 2017, JCoPh, 335, 84
Krivodonova, L. 2007, JCoPh, 226, 879
Kuroda, T., Takiwaki, T., & Kotake, K. 2016, ApJS, 222, 20
Laiu, M. P., Harris, J. A., Chu, R., & Endeve, E. 2020, JPhCS, 1623, 012013
Lalazissis, G. A., König, J., & Ring, P. 1997, PhRvC, 55, 540
Larsen, E. W., & Morel, J. E. 1989, JCoPh, 83, 212
Larson, M. G., & Bengzon, F. 2013, The Finite Element Method: Theory, Implementation, and Applications (Berlin: Springer)
Lattimer, J. M., Pethick, C. J., Ravenhall, D. G., & Lamb, D. Q. 1985, NuPhA, 432, 646
Lattimer, J. M., & Swesty, D. F. 1991, NuPhA, 535, 331
Lax, P. D., & Liu, X.-D. 1998, SIAM J. Sci. Comput., 19, 319
LeVeque, R. 1992, Numerical Methods for Conservation Laws (Berlin: Springer)
Li, G., & Xing, Y. 2018, JCoPh, 352, 445
Liu, X. D., & Osher, S. 1996, SJNA, 33, 760
Mabanta, Q. A., & Murphy, J. W. 2018, ApJ, 856, 22
Martínez-Pinedo, G., Fischer, T., & Huther, L. 2014, JPhG, 41, 044008
Melson, T., Kresse, D., & Janka, H.-T. 2020, ApJ, 891, 27
Mezzacappa, A. 2001, NuPhA, 688, 158
Mezzacappa, A. 2005, ARNPS, 55, 467
Mezzacappa, A., & Messer, O. 1999, JCoAM, 109, 281
Miller, J. M., & Schnetter, E. 2017, CQGra, 34, 015003
Mönchmeyer, R., & Müller, E. 1989, A&A, 217, 351
Müller, B. 2020, LRCA, 6, 3
Müller, B., Heger, A., Liptai, D., & Cameron, J. B. 2016, MNRAS, 460, 742
Müller, B., Janka, H.-T., & Dimmelmeier, H. 2010, ApJS, 189, 104
Müller, B., Janka, H.-T., & Heger, A. 2012, ApJ, 761, 72
Müller, B., & Varma, V. 2020, MNRAS, 498, L109
Murphy, J. W., Dolence, J. C., & Burrows, A. 2013, ApJ, 771, 52
Murphy, J. W., & Meakin, C. 2011, ApJ, 742, 74
Nagakura, H., Sumiyoshi, K., & Yamada, S. 2014, ApJS, 214, 16
O'Connor, E. P., & Couch, S. M. 2018, ApJ, 865, 81
Oertel, M., Hempel, M., Klähn, T., & Typel, S. 2017, RvMP, 89, 015007
Olbrant, E., Hauck, C. D., & Frank, M. 2012, JCoPh, 231, 5612
Omang, M., Børve, S., & Trulsen, J. 2006, JCoPh, 213, 391
Ott, C. D., Schnetter, E., Burrows, A., et al. 2009, JPhCS, 180, 012022
Qiu, J., & Shu, C.-W. 2005, SIAM J. Sci. Comput., 27, 995
Quirk, J. J. 1994, IJNMF, 18, 555
Radice, D., Abdikamalov, E., Ott, C. D., et al. 2018, JPhG, 45, 053003
Radice, D., Couch, S. M., & Ott, C. D. 2015, ComAC, 2, 7
Radice, D., Ott, C. D., Abdikamalov, E., et al. 2016, ApJ, 820, 76
Radice, D., & Rezzolla, L. 2011, PhRvD, 84, 024010
Rampp, M., & Janka, H. T. 2002, A&A, 396, 361
Reed, W., & Hill, T. 1973, Report LA-UR-73-479 (Los Alamos, NM: Los Alamos Scientific Laboratory)
Remacle, J.-F., Flaherty, J. E., & Shephard, M. S. 2003, SIAMR, 45, 53
Rezzolla, L., & Zanotti, O. 2013, Relativistic Hydrodynamics (Oxford: Oxford Univ. Press)
Rivière, B. 2008, Discontinuous Galerkin Methods for Solving Elliptic and Parabolic Equations Theory and Implementation, Frontiers in Applied Mathematics (Philadelphia, PA: SIAM), 35
Roberts, L. F., Ott, C. D., Haas, R., et al. 2016, ApJ, 831, 98
Schaal, K., Bauer, A., Chandrashekar, P., et al. 2015, MNRAS, 453, 4278
Schneider, A. S., Roberts, L. F., Ott, C. D., & O'Connor, E. 2019, PhRvC, 100, 055802
Shen, G., Horowitz, C. J., & O'Connor, E. 2011a, PhRvC, 83, 065808
Shen, G., Horowitz, C. J., & Teige, S. 2011b, PhRvC, 83, 035802
Shen, H., Toki, H., Oyamatsu, K., & Sumiyoshi, K. 1998, PThPh, 100, 1013
Shu, C.-W. 2016, JCoPh, 316, 598
Shu, C.-W., & Osher, S. 1988, JCoPh, 77, 439
Skinner, M. A., Dolence, J. C., Burrows, A., Radice, D., & Vartanyan, D. 2019, ApJS, 241, 7
Sod, G. A. 1978, JCoPh, 27, 1
Soderberg, A. M., Chakraborti, S., Pignata, G., et al. 2010, Natur, 463, 513
Steiner, A. W., Hempel, M., & Fischer, T. 2013a, ApJ, 774, 17
Steiner, A. W., Lattimer, J. M., & Brown, E. F. 2010, ApJ, 722, 33
Steiner, A. W., Lattimer, J. M., & Brown, E. F. 2013b, ApJL, 765, L5
Stone, J. M., & Norman, M. L. 1992, ApJS, 80, 753
Sugahara, Y., & Toki, H. 1994, NuPhA, 579, 557
Sumiyoshi, K., & Yamada, S. 2012, ApJS, 199, 17
Summa, A., Hanke, F., Janka, H.-T., et al. 2016, ApJ, 825, 6
Suresh, A., & Huynh, H. T. 1997, JCoPh, 136, 83
Swesty, F. D. 1996, JCoPh, 127, 118
Tamborra, I., Hanke, F., Janka, H.-T., et al. 2014, ApJ, 792, 96
Teukolsky, S. A. 2016, JCoPh, 312, 333
Timmes, F. X., & Swesty, F. D. 2000, ApJS, 126, 501
Todd-Rutel, B. G., & Piekarewicz, J. 2005, PhRvL, 95, 122501
Toro, E. F., Spruce, M., & Speares, W. 1994, ShWav, 4, 25
Turk, M. J., Smith, B. D., Oishi, J. S., et al. 2011, ApJS, 192, 9
Vartanyan, D., Burrows, A., & Radice, D. 2019, MNRAS, 489, 2227
Vincent, T., Pfeiffer, H. P., & Fischer, N. L. 2019, PhRvD, 100, 084052
Virtanen, P., Gommers, R., & Oliphant, T. E. 2020, NatMe, 17, 261
Wilson, J. R., Mathews, G. J., & Marronetti, P. 1996, PhRvD, 54, 1317
Woosley, S. E., & Heger, A. 2007, PhR, 442, 269
Wu, K., & Tang, H. 2015, JCoPh, 298, 539
Xing, Y., Zhang, X., & Shu, C.-W. 2010, AdWR, 33, 1476
Yahil, A. 1983, ApJ, 265, 1047
Zhang, W., Almgren, A., Beckner, V., et al. 2019, JOSS, 4, 1370
Zhang, X., & Shu, C.-W. 2010a, JCoPh, 229, 8918
Zhang, X., & Shu, C.-W. 2010b, JCoPh, 229, 3091
Zhang, X., & Shu, C.-W. 2011, RSPSA, 467, 2752
Zhu, J., Qiu, J., & Shu, C.-W. 2020, JCoPh, 404, 109105
Zingale, M., & Katz, M. P. 2015, ApJS, 216, 31